GENETIC ALGORITHMS FOR SOLVING SCHEDULING PROBLEMS IN MANUFACTURING SYSTEMS

Anna ŁAWRYNOWICZ

Faculty of Management
Warsaw University of Technology, Warsaw, Poland
a.lawrynowicz@wz.pw.edu.pl

***Abstract*:** Scheduling manufacturing operations is a complicated decision making process. From the computational point of view, the scheduling problem is one of the most notoriously intractable NP-hard optimization problems. When the manufacturing system is not too large, the traditional methods for solving scheduling problem proposed in the literature are able to obtain the optimal solution within reasonable time. But its implementation would not be easy with conventional information systems. Therefore, many researchers have proposed methods with genetic algorithms to support scheduling in the manufacturing system. The genetic algorithm belongs to the category of artificial intelligence. It is a very effective algorithm to search for optimal or near-optimal solutions for an optimization problem. This paper contains a survey of recent developments in building genetic algorithms for the advanced scheduling. In addition, the author proposes a new approach to the distributed scheduling in industrial clusters which uses a modified genetic algorithm.

***Keywords*:** manufacturing system, scheduling, genetic algorithm, genetic algorithms for the advanced.

1 Introduction

Scheduling problem is an assignment problem, which can be defined as the assigning of available resources (machines) to the activities (operations) in such a manner that maximizes the profitability, flexibility, productivity, and performance of a production system (Prakash et al. [49). Scheduling of operations is one of the most critical issues in the planning and managing of manufacturing processes. The literature review indicates that meta-heuristics may be used for the advanced scheduling in manufacturing systems and the genetic algorithm is one of the meta-heuristics that has attracted many researchers. Therefore, the main objective of this paper is to present a survey of the recent developments of evolutionary-based methods for the advanced scheduling. The author has to arbitrarily select the most representative work known to them, because it is impossible to provide an exhaustive literature review discussing every piece of work that has been done over the years.

The survey is structured in the following way. At the beginning, an introduction to evolutionary algorithms is presented in Section 2. Section 3 contains an overview of recent developments in building the genetic algorithms for the advanced scheduling. This survey categorizes the literature according to shop environments, including parallel machines, flow shop, permutation flow shop, job shop, flexible job shop, open job shop, and others. In Section 4, the author proposes a new approach to the distributed scheduling in industrial clusters which uses a modified genetic algorithm. Finally, a discussion on the current research status and most promising paths of future research is presented in Section 5.

2 Genetic algorithm

The genetic algorithm (GA) belongs to the category of evolutionary algorithm. Evolutionary Algorithms (EAs), a class of heuristic search techniques inspired to survival-of-the-fittest Darwinian evolution principles, work iteratively on a population of candidate solutions of the given problem. The Darwinian metaphor is transformed in a stochastic search algorithm in which genetic crossover, mutation and selection processes are emulated with specific mathematical operators. Unlike some other efficient meta-heuristics, EAs are flexible and therefore they have been successfully applied to many single and multi-objective optimization problems.

Evolutionary algorithms have three instantiations: genetic algorithms (GAs), evolutionary programming (EP), and evolution strategies (ESs). Among them, genetic algorithms are probability the most well known and widely used (Guang and Hong [26]).

Genetic algorithms are probabilistic search algorithms, which mimic biological evolution to produce gradually

better offspring solutions (Ying-Hua and Young-Chang [59]). Each solution to a given problem can be encoded by a chromosome that represents an individual in a population. Each chromosome is made up of a sequence of genes from a certain alphabet. The alphabet can be a set of binary numbers, real numbers, integers, symbols, or matrices (Goldberg [25]). The representation scheme determines not only how effective the problem is structured, but also how efficient the genetic operators can be used. The population is evolved, over generations, to produce better solution to the problem. The evolution of the genetic algorithm population from one generation to the next is usually achieved through the use of three operators that are fundamental in GA: selection, crossover, and mutation. The cycle of evaluation-selection-reproduction is continued until a termination criterion is reached. J.H. Holland in 1975 first described a GA, which is commonly called the classical genetic algorithm (CGA).

2.1 Procedure of the classical genetic algorithm

The overall procedure of the classical genetic algorithm is outlined below.

Procedure: Classical genetic algorithm (CGA)

```
Begin:
    t ← 0;
    initialise population P(t);
    evaluate P(t);
        While (not termination condition) do
        Begin
            t ← t + 1
            select P(t) from P(t - 1)
            recombine P(t) by crossover and mutation;
            evaluate P(t);
        End;
End.
```

The genetic parameters, namely number of generation, probability of crossover, probability of mutation, are optimised relating to the size of problems.

In general, there are need the following basic components to implement an evolutionary algorithm in order to solve a problem (Carlos and Coello [6]):

1) a representation of the potential solutions to the problem;
2) a way to create an initial population of potential solutions (this is normally done randomly, but deterministic approaches can also be used);
3) an evaluation function that plays the role of the environment, rating solutions in terms of their ‘‘fitness’’;
4) a selection procedure that chooses the parents that will reproduce;
5) evolutionary operators that alter the composition of children (normally, crossover and mutation);
6) values for various parameters that the evolutionary algorithm uses (population size, probabilities of applying evolutionary operators, etc.).

2.2 Representation

In ordering problem using the genetic algorithm, critical issue is developing a representation scheme to represent a feasible solution. As mentioned above genetic algorithms work with a population of potential solution to a problem. A population is composed of chromosomes (i.e. a string), where each chromosome represents one potential solution. Traditional binary vectors used to represent the chromosome are not effective in such a large-scale dimension. During the last years, the following nine representations for the job-shop scheduling problem have been often proposed: operation-based representation, job-based representation, preference list-based representation, job pair relation-based representation, priority rule-based representation, disjunctive graph-based representation, completion time-based representation, machine-based representation, random keys representation and others. A tutorial survey of job shop scheduling problem using different representation in genetic algorithm has been published by Cheng et al. [13].

The most popular encoding methods are operation-based representation, job-based representation and random keys representation which are presented below.

- Operation-based representation

In the scheduling problem, the popular representation is operation-based method. This representation encodes a schedule as a sequence of operations and each gene stands for one operation. One natural way to name each operation is using a natural number. A schedule is decoded from a chromosome with the following decoding procedure (Cheng et al. [13]): (a) firstly translate the chromosome to a list of ordered operations; (b) then generate the schedule by a one-pass heuristic based on the list. The first operation in the list is scheduled first, then the second operation, and so on. Each operation is allocated in the best available time for the corresponding machine the operation requires.

Table 1. Example of 3-jobs and 3-machines
(source: own study)

Job	1			2			3		
Operation	1	2	3	1	2	3	1	2	3
Processing time	2	5	3	4	3	2	2	3	4
Machine	1	2	1	3	1	2	2	3	3

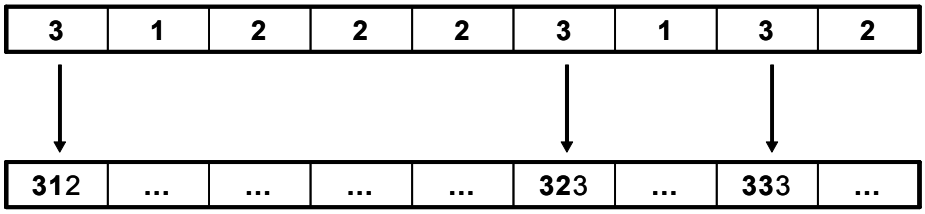

Figure 1. Operation-based representation
(source: own study)

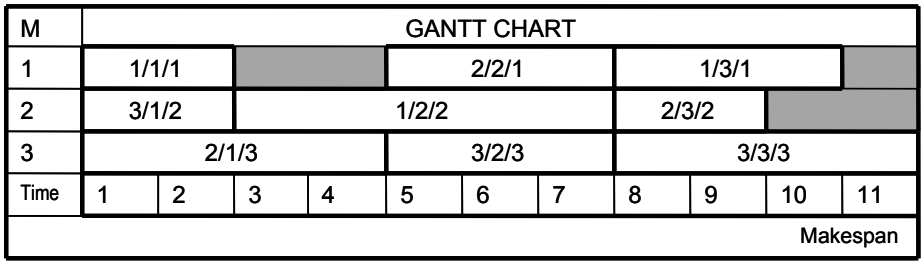

Figure 2. Decoded active schedule
(source: own study)

The process is repeated until all operations are scheduled. As an example, consider the 3-job 3-machine problem given in Table 1. Suppose a chromosome is given as [3 1 1 2 2 3 1 3 2]. Each gene uniquely indicates an operation, and can be determined according to the order of occurrence in the sequence (see Fig. 1). Let o_{jim} denote the ith operation of job j on machine m. The chromosome can be translated into a unique list of ordered operations of [o_{312} o_{111} o_{122} o_{213} o_{221} o_{323} o_{131} o_{333} o_{232}]. Operation o_{312} has the highest priority and is scheduled first, then o_{111} , and so on. The resulting active schedule is shown in Fig. 2

- Job-based representation

The popular encoding method is also the job-based representation. This representation consists of a list of n jobs and a schedule is constructed according to the sequence of jobs. For a given sequence of jobs, all operations of the first jobs in the list are scheduled first, and then the operations of second job in the list are considered. The first operation of the job under treatment is allocated in the best available processing time for the corresponding machine the operation requires, and then the second operation, and so on until all operations of job are scheduled. The process is repeated with each of the jobs in the list considered in the appropriate sequence.

Consider the 5-job 3-machine problem given in Table 2. Suppose a chromosome is given as [5 4 2 3 1]. The first job to be processed is job 5. The operation precedence constraint for job 2 is [m_1 m_2 m_3] and the corresponding processing time for each machine is [2 3 4]. Firstly, the operations of job 5 are scheduled. Then the job 4 is processed, its operations precedence among machines is [m_3 m_1 m_2] and the corresponding processing time for each machine is [2 4 2]. Next, the jobs 2, 3 are processed. Lastly, the operations of job 1 are scheduled as shown in Fig. 3.

Table 2 Example of 5 job on 3 machine
(*source: own study)*

Job	1			2			3			4			5		
Operation	1	2	3	1	2	3	1	2	3	1	2	3	1	2	3
Processing time	2	2	2	2	3	2	2	2	2	2	4	2	2	3	4
Machine	3	1	2	3	1	2	2	1	3	3	1	2	1	2	3

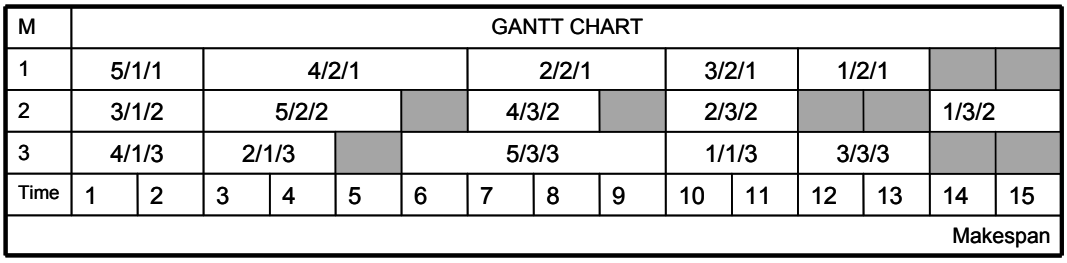

Figure 3. Jobs scheduling
(*source: own study)*

- Random key representation

Random key representation encodes a solution with random number (Chen and Ji [11]). These values are used as sort keys to decode the solution. For n-job m-machine scheduling problem, each gene (a random key) consists of two parts: an integer in set {1, 2, …, m} and a fraction generated randomly from {0, 1}. The integer part of any random key is interpreted as the machine assignment for that job. Sorting the fractional parts provides the job shop sequence on each machine.

The rest of this paper presents a brief review of the literature which includes different encoding methods.

2.3 Fitness function

The newly created individuals are evaluated and assigned fitness values. Then, either all or only a subset of the current population is replaced by these new individuals. Thus, the evaluation includes compute fitness value, which is a measure of how well the individual optimizes the function. Test each individual uses the objective function. In other word, the fitness value is used to determine the selection probability for each chromosome. In proportional selection procedure, the selection probability of a chromosome is proportional to its fitness value. Hence, filter chromosomes have higher probabilities of being selected to next generation.

2.4 Crossover operator

Crossover is an operation to generate a new chromosome (i.e. child or offspring) from two parents. It is the main operator of GA. During the past years, various crossover operators had been proposed such as partial-mapped crossover (PMX), order crossover (OX), cycle crossover (CX), position-based crossover, etc. The two most popular crossover operators are partial mapped crossover (Liaw [32], Moon et al. [42]) and order crossover [Arroyo and Armentano [2], França et al. [18], Jolai et al. [29]).

- Partial mapped crossover

Partial mapped crossover was proposed by Goldberg and Lingle [24]. It can be viewed as a variation of two-cut-point crossover that incorporates a special repairing procedure to resolve possible illegitimacy. PMX has the following major steps:

1) select two cut-points along the string at random; the substrings defined by the two cut-points are called the mapping sections;
2) exchange two substrings between parents to produce proto-children;
3) determine the mapping relationship between two mapping sections;
4) legalize offspring with the mapping relationship.

- Order crossover

Order crossover was proposed by Davis (17). It can be viewed as a kind of variation of PMX that uses a different repairing procedure. OX has the following major steps:

1) select a substring from one parent at random;
2) produce a proto-child by copying the substrings into the corresponding positions as they are in the parent;
3) delete all the symbols from the second parent, which are already in the substring; the resultant sequence contains the symbols the proto-child needs;
4) place the symbols into the unfixed positions of the proto-child from left to right according to the order of the sequence to produce an offspring.

A survey of order crossovers can be found in the work of Cheng et al. [14].

2.5 Mutation

Mutation is used to produce perturbations on chromosomes in order to maintain the diversity of population. In the literature, two main types of mutation operators named inversion mutation and insertion mutation are used (Zhang et al. [61]). Inversion mutation serves to maintain the diversity in population. Insertion mutation is used not only to produce small perturbations but also to perform intensive search in order to find an improved offspring. Inversion mutation and insertion mutation act on half of the population, respectively. The mutations are described as follows:

- inversion mutation inverts the substring between two different random positions,
- insertion mutation selects two elements randomly and inserts the back one before the front one.

2.6 Selection

Selection is another important factor to consider in implementing GA. It is a procedure to select offspring from parents to the next generation. According to the general definition, the selection probability of a chromosome should show the performance measure of the chromosome in the population. Hence a parent with a higher performance has higher probabilities of being selected to next generation (Chen et al. [12]).

In the reproduction operation, there are two kind of well-known selection mechanisms: the roulette wheel selection (Chen and Ji [11]) and tournament selections (Vallada and Ruiz [57]).

- Roulette wheel selection

The roulette wheel selection can be visualized by imagining a wheel where each chromosome occupies an area that is related to its value of objective function. When a spinning wheel stops, a fixed marker determines which chromosome will be selected to reproduce into the mating pool (Blanco et al. [4]). Such a selection mechanism needs more numerical computations.

- Tournament selection

The tournament selection is quite simple and suitable for checking whether a chromosome is reproduced or not according to its corresponding objective function. In the tournament selection, p_r x N chromosomes with minimum objective functions are more added into the population, and correspondingly p_r x N chromosomes with maximum objective functions are discarded from the population. The population still keeps the same size (Chang [9]).

3 Application of genetic approach for advanced scheduling

Scheduling plays an important role to implement effective operations management methods. But its implementation would not be easy with the conventional information systems (Chang [9]). During the past few decades, genetic algorithms have received a lot of attention regarding their potential as global optimization techniques for complex optimization problems. Therefore, a short literature review on the adaptation of genetic algorithms to manufacturing operations is presented below.

3.1 Parallel machines scheduling problem

The problem can be described as follows: there are m machines in parallel where machines may be identical, or have different speeds or uniform, or completely unrelated. Each job can be performed on any of the machines (Allahverdi [1]). Several approaches were proposed to solve this kind of problems. For example, to solve the parallel machines scheduling problem a two-phase sub-population genetic algorithm was proposed by Chang et al. [10]. The algorithm is divided into two phases. The first phase applies subpopulations, which concentrates on specific search space and prevents all individuals from converging to a local optimal. Then, in order to explore the solution space

ignored or missed in the first phase, sub-populations are regrouped as a single big population. Each individual chromosome in this big population of the second phase is randomly assigned a weight value to explore more of the solution space. Experimental results are reported and the superiority of this approach is discussed.

To solve the multiobjective scheduling model on parallel machines (MOSP), a new parallel genetic algorithm (PIGA) based on the vector group encoding method and the immune method was proposed by Gao et al. [20]. Compared with other scheduling problems on parallel machines, the MOSP is distinct for the following characteristics:

1) parallel machines are nonidentical;
2) the type of jobs processed on each machine can be restricted;
3) the multiobjective scheduling problem includes minimizing the maximum completion time among all the machines (makespan) and minimizing the total earliness/tardiness penalty of all the jobs.

For PIGA, its three distinct characteristics are as follows: Firstly, individuals are represented by a vector group, which can effectively reflect the virtual scheduling policy. Secondly, an immune operator is adopted and studied in order to guarantee diversity of the population. Finally, a local search algorithm is applied to improve the quality of the population. Numerical results show that it is efficient, can better overcome drawbacks of the general genetic algorithm, and has better parallelism.

A new encoding method in order to adapt the GA to non-identical parallel machine scheduling problem was also proposed by Balin S. [3]. The encoding method is as follows: The raw i of the matrix X consists of jobs to be processed on machine i. Raws are called ''genes'' (g_1, ... g_i, ... g_m) and they represent jobs to be processed on each machine; jobs to be processed on machine i are given by elements non-zero of gene i ($x(i, j) = 1$). The completion time of each machine i, (C_i), is equal to the sum of processing times of jobs to be processed on that machine; it is called as the ''value of gene i'' and it is defined by the following function:

$$f(g_i) = \sum_{j=1}^{n} x(i,j) \times P(i,j), \quad i = 1,..., m \tag{1}$$

In last years, a hybrid memetic algorithm for maximizing the weighted number of just-in-time jobs on unrelated parallel machines was also presented by Jolai et al. [29]. Unrelated parallel machines can be characterized as machines that execute the same function but have different industrial unit may invest in related machines. A memetic algorithms (MA) is a genetic algorithm hybridized with a local search (LS) procedure used to intensify the search process (Jolai et al. [29]). Besides, in the literature, a genetic algorithm for the unrelated parallel machine scheduling problem with sequence dependent setup limes was reported by Vallada and Ruiz R. [57].

3.2 Permutation flow shop scheduling problem

The general flow shop scheduling problem is a production problem where a set of n jobs have to be processed with identical flow pattern on m machines. In permutation flow shops the sequence of jobs is the same on all machines (Nagano et al. [44]). In other words, the permutation flowshop scheduling problem (PFSP) consists in scheduling a set of n jobs on m machines in the same technological order, such that each job is processed on machine 1 in the first place, machine 2 in the second place, ... and machine m in the last place; the processing time of job i on machine j is denoted p_{ij}.

The most frequently used encoding for the PFSP is a simple permutation of the jobs. It is important to note that there are several additional conditions to this problem (Ruiz et al. [53]):

- all operations are independent and available for processing at time 0,
- all m machines are continuously available,
- each machine i can process at most one job j at a time,
- each job j can be processed only on one machine i at a time.

The objective is to find a sequence (schedule) in which these n jobs should be processed on each of the m machines such that a given criterion be optimized (Jarboui et al. [27]). The most common criteria are the minimization of the total completion time of the schedule often referred to as makespan C_{max} and the total flow time minimization. The PFSP has been extensively investigated by the research community. For example, a genetic local search algorithm for minimizing total flow time in the permutation flow shop scheduling problem was developed by Tseng and Lin [55] and Xu et al. [58]. The permutation flowshop scheduling problem with the objective of minimizing makespan was presented by Nearchou [45], Rajkumar and Shahabudeen [51], Nagano et al. [44], and Ruiz et al. [52, 53].

An effective memetic algorithm for solving multiobjective permutation flow shop scheduling problems with minimization of makespan and total flow time was considered by Chiang et al. [15]. The results of above mentioned study show that the proposed algorithms are very effective.

3.3 Flow shop scheduling problem

In general, the flow-shop scheduling problem (FSSP) is a strongly NP-hard combinatorial optimization problem that has captured the interest of a significant number of researchers (Nearchou [45]). The flow-shop scheduling problem is one of the most well known problems in the area of scheduling. It is a production planning problem in which n jobs have to be processed in the same sequence on m machines. Most of these problems concern the objective of minimizing makespan i.e. the time between the beginning of the execution of the first job on the first machine and the completion of the execution of the last job on the last machine. To minimize the makespan is equivalent to maximize the utilization of the machines (Chen et al. [12]).

The flowshop scheduling problem has been widely studied in the literature and many techniques for its solution have been proposed. Many authors have concluded that genetic algorithms are suitable for this hard, combinatorial problem. For example, the flowshop scheduling problem with the objective of minimizing makespan was considered among other things by França et al. [18], Ruiz and Maroto [52, 53], Kim and Jeong [30], Rajkumar and Shahabudeen [51], Liao and Tsai [34].

França et al. [18] suggested an evolutionary algorithm for scheduling a flowshop manufacturing cell with sequence dependent family setups. They proposed evolutionary heuristic algorithms to minimize the makespan in a pure flow shop manufacturing cell problem with sequence dependent setup times between families of jobs. The heuristic algorithms implemented are a memetic algorithm, a genetic algorithm and a multi-start strategy. Computational results show that the proposed algorithms are relatively more effective in minimizing the makespan than the best known heuristic algorithm. The flow shop scheduling problem with the objective of minimizing makespan was also developed by Ruiz and Maroto [52]. They developed a genetic algorithm for hybrid flow shops with sequence dependent setup times and machine eligibility. Numerical computation based on benchmarks demonstrated the effectiveness of the proposed method. An improved genetic algorithm with the objective of minimizing the makespan for the flow shop scheduling problem was also proposed by Rajkumar and Shahabudeen [51].

Recently, some genetic algorithms have been developed for the multi-objective flow shop problem. For example, Arroyo and Armentano [2] presented a multi-objective genetic local search algorithm, which was applied to multi-objective flow shop problems in order to find an approximation of the Pareto optimal set. The algorithm is applied to the flow shop scheduling problem for the following two pairs of objectives: (i) makespan and maximum tardiness; (ii) makespan and total tardiness. Computational results show that the proposed algorithm yields a reasonable approximation of the Pareto optimal set. Beside, Onwubolu and Davendra [47] developed a differential evolution algorithm for the flow shop scheduling problem in which makespan, mean flowtime, and total tardiness are the performance measures.

3.4 Job shop scheduling problem

The job shop scheduling problem (JSP) is well known as one of the most complicated combinatorial optimization problems, and it is a NP-hard problem (Gao et al. [22]).

The general job shop scheduling problem (JSP) with the makespan criterion can be described by a set of n jobs that must be processed on m machines. Each job composes of several operations, and the operations of a given job have to be processed in a given order. Each operation uses one of the m machines for a fixed duration. Each machine can process at most one operation at a time, and once an operation initiates processing on a given machine, it must complete processing on that machine without interruption (Zhang et al. [60]). In general, the objective is to find the optimal schedule of the operations on the machines, taking into account the precedence constraints, which minimizes the makespan, i.e., the finish time of the last operation completed in the schedule (Gao et al. [19]).

Many different approaches have been applied to JSP and a rich harvest has been obtained. The most important part of the literature concerning job shop scheduling problems is dedicated to single-criterion optimization. But, in practice, the use of multiple criteria often enables one to compute more realistic solutions for a decision maker working in production

planning. For this reason several works have recently tackled multi-objective job shop scheduling problems. A genetic algorithm for a multi-objective job shop scheduling problem that minimizes the mean weighted completion time and the sum of the weighted tardiness/earliness costs simultaneously was developed by Tavakkoli-Moghaddam et al. [54].

Besides, an efficient memetic algorithm for solving the job shop scheduling problem can be found in the work of Gao et al. [22].

Among various kinds of encoding methods, job-based encoding (Zhang and Wu [62]) and operations-based encoding (Zhang et al. [61]) are most often used for job shop scheduling problem. A genetic algorithm with new encoding scheme for job shop scheduling was developed by Wang et al. [56]. They proposed a novel genetic chromosome-encoding approach. In this encoding method, the operation of crossover and mutation was done in three-dimensional coded space. Some big benchmark problems were tried with the proposed three-dimensional encoding genetic algorithm for validation and the results are encouraging.

A genetic algorithm for job shop scheduling problems with alternative routings was also proposed by Moon et al. [42]. In this approach, the chromosome is composed of two parts. The first part is for the assignment of alternative machines, and the second part is the relative processing order between jobs. The length of each chromosome is equal to the total number of operations. This genetic algorithm generated relatively good solutions quickly.

3.5 Flexible job shop scheduling problem

Genetic algorithms are also used as an optimization tool for solving the flexible job-shop scheduling problem (FJSP). Flexible job shop scheduling problem is an extension of the classical job shop scheduling problem, which provides a closer approximation to a wide range of real manufacturing systems. In particular, there are a set of work centers in a flexible job shop. Each work center has a set of parallel machines with possibly different efficiency. An operation can be performed by any machine in a work center. Consequently, this results in two problems. The first one is the routing problem (i.e., the assignment of operations to machines), and the second one is the scheduling problem (i.e., determining the starting time of each operation).

The combination of the two decisions presents additional complexity and a new problem called flexible job shop scheduling problem (FJSP) (Gao et al. [19]).

In the flexible job-shop scheduling problem, the objective is usually to minimize the makespan. For example, Zhang et al. [60] proposed an effective genetic algorithm for the flexible job-shop scheduling problem with the minimization of makespan.

Recently, some genetic algorithms have been developed for the multi-objective flexible job-shop scheduling problems. For example, Gholami and Zandieh [23] proposed a genetic algorithm where the objectives are the minimization of two criteria, the makespan and the mean tardiness.

The current work pursues research in which GA procedure is combined with experts' knowledge. FMS scheduling with knowledge based genetic algorithm (KBGA) was reported by Prakash et al. [49]. The KBGA is a stochastic search technique with the inherent ability of GA and strength of knowledge to enhance the performance of system and algorithm concurrently. In this study, two objective functions known as throughput and mean flow time, have been taken to measure the performance of the FMS.

In genetic algorithms for the flexible job-shop scheduling problem, many different representations are used. For example, Gao et al. [21] proposed encoding method where every chromosome consists of a machine assignment vector V_1 and an operation sequence V_2. In this case, $V_1(r)$ represents the machine chosen to process the operation indicated at position r. The authors identify all operations of a job with the same sign; then, they interpret the signs according to the order of occurrences in the sequence of a given chromosome; therefore, each job i appears in the operation sequence vector (V_2) exactly n_i times to represent its n_i ordered operations. The encoding method was adopted by Gholami and Zandieh [23] to schedule a dynamic flexible job shop with genetic algorithm. The same components i.e. machine selection and operation sequence (called MSOS), include the chromosome representation were proposed by Zhang et al. [60] (see Fig. 4).

	Machine Selection (MS)					Operation Sequence (OS)				
Chromosome =	4	1	2	2	4	2	2	1	1	2

Figure 4. Structure of proposed MSOS chromosome *(source: Zhang et al. [60])*

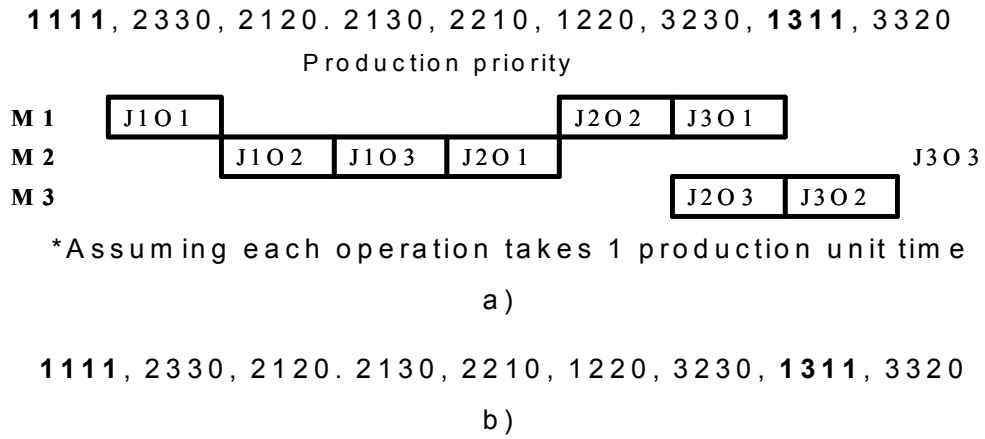

Figure 5. (a) A sample encoding and decoding of chromosome
(b) A sample encoding of alternative routing
(*source: Chan et al. [8]*)

An idea, namely genetic algorithms with dominant genes (GADG) in order to deal with FMS scheduling problems with alternative production routing was developed by Chan et al. [8]. It consists of ΣN_i genes, and each gene consists of four parameters representing machine, job, operation, and domination (MJOD).

Fig. 5a shows a sample encoding of a chromosome for the scheduling of three jobs on three machines, and each job possesses three operations with a total of nine genes. In Fig. 5a, the second gene (2330) represents that O3 of J3 is allocated on M2. It is not a DG as the D parameter denoted by 0, otherwise it will be denoted by 1.

The production priority of jobs on machines is defined by the ordering, from the highest priority on the left to the lowest on the right. In this connection, O3 of J3 (the second gene: 2330) is scheduled before O2 of J1 (the third gene: 2120) on M2. However, since an operation can only start after its preceding operation is completed, O3 of J3 will not be considered until O2 of J3 is finished. In this situation, the third gene (2120) O2 of J1 will be scheduled for production instead.

A detailed production schedule is shown in figure 5a. In an FMS environment, assuming O3 of J3 can also be performed on M3, the second gene can be represented as (3330) as shown in Fig. 5b.

An improved memetic algorithm to solve the job shop scheduling problem was also proposed in work of Gao et al. [22]. As mentioned above, the memetic algorithm is a genetic algorithm hybridized with a local search procedure used to intensify the search process. The flow chart of the proposed MA by Gao et al. [22] is shown in Fig. 6. The procedure of the MA is outlined as follows:

- Step 1

Generate initial population. Set parameters of GA including population size, max iteration, mutation probability, crossover probability, etc. Then encode an initial solution into a chromosome. Repeat this step until the number of individual equals to the population size.

- Step 2

Apply the local search procedure to improve the quality of each individual.

- Step 3

Decode each individual of population to obtain the makespan corresponding with each individual. And compare them to obtain the best solution.

- Step 4

Check the termination criteria. If one of the criteria is satisfied, then stop the algorithm and output the best solution; otherwise, go to step 5.

- Step 5

Generate new population for the next generation. Genetic evolution with three operators including selection, crossover and mutation is applied to create offspring for the next population. Following this, the algorithm goes back to step 2.

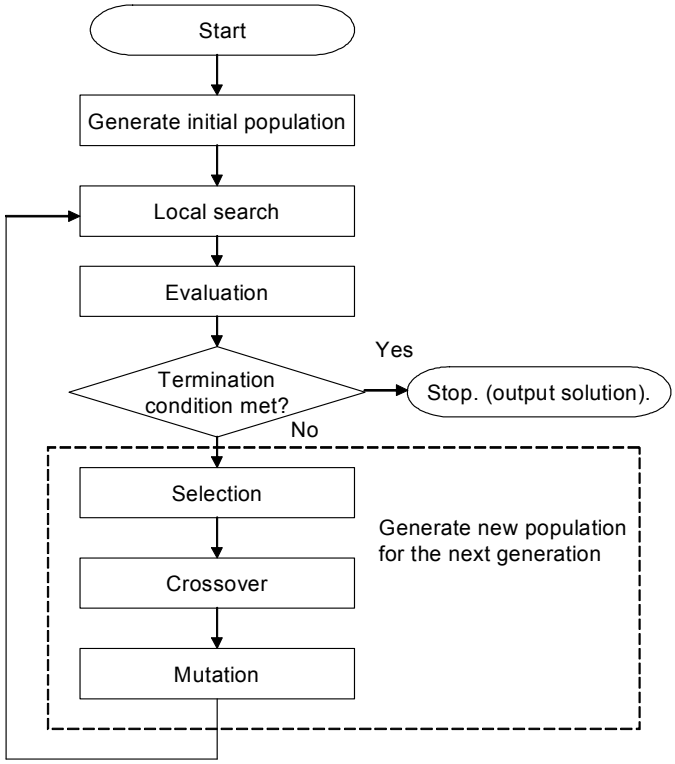

Figure 6. Flow chart of the MGA
(source: Gao et al. [22])

3.6 Open job shop scheduling problem

An open shop scheduling problem (OSSP) can be stated as follows: There are n jobs to be processed on m machines. Each job consists of m operations where each operation can be done on only one of machines for a give process time. Each operation can be processed on at most one machine at any time. On each machine at any time at most one operation can be done. The OSSP is the same as job shop scheduling problem (JSP), except there is no precedence relation between operations in the OSSP. In this way there will be more feasible combinations in the OSSP (Panahi and Tavakkoli-Moghaddam [48]).

Low and Yeh [35] developed a genetic algorithm-based heuristics for an open shop scheduling problem with setup, processing, and removal times separated.

Their report proposes a solution to the open shop scheduling problem with the objective of minimizing total job tardiness in the system. In open job shop scheduling problem, there are two essential issues addressing all kinds of open shop scheduling problems: determining the routing for each job, and sequencing the jobs for each machine.

Adequately, a permutation representation is presented to encode these two things into a chromosome. This representation encodes a schedule as an ordered sequence of job-machine combinations (operations), where each gene in a chromosome stands for an operation. In this representation, operations are listed in the order in which they are scheduled. A chromosome is merely a permutation of the number from 1 to the total number of operations to be scheduled in the system.

Consider a simple example with three jobs and three machines. Each job must be processed on every machine once; operations of a job can be processed in any order. A series of numbers from 1 to 9 is assigned to, each job-machine combination, as in Table 3. Thus a chromosome [6-3-1-4-9-7-2-8-5] can be decoded to routing for each job and a processing sequence on each machine, respectively. A feasible schedule is then determined as follows:

Machine #1: Job 1-Job 2-Job 3
Machine #2: Job 1-Job 3-Job 2
Machine #3: Job 2-Job 1-Job 3

Table 3 Representation of job-machine combination
(source: Low and Yeh [35])

Job	1	1	1	2	2	2	3	3	3
Machine	1	2	3	1	2	3	1	2	3
Number	1	2	3	4	5	6	7	8	9

3.7 Hybrid approaches

Currently, there is a research trend in the adaptation of hybrid approaches which combine different concepts or components of various techniques. The trends have been presented by Kobbacy et al. [31] in a very interesting survey of applications of artificial intelligence techniques for operations management.

They reported that several authors use genetic algorithms to carry out an intelligent search by proposing alternative schedules and then using neural network to asses the quality and fitness of the schedule. Besides, fuzzy logic and genetic algorithms have been combined effectively for scheduling.

A hybrid genetic algorithm was developed by Chen et al. [12] for the re-entrant flow-shop scheduling problem (RFS). In a RFS, all jobs have the same routing over the machines of the shop and the same sequence is traversed several times to complete the jobs. The aim of this study was to minimize the makespan by using the genetic algorithm (GA) to move from the local optimal solution to the near optimal solution for RFS scheduling problems.

For the job shop scheduling problem, a hybrid evolutionary algorithm was also presented in work of Zobolas et al. [64]. In their work, the optimization criterion is minimization of the makespan and the solution method consists of three components: a Differential Evolution-based algorithm to generate a population of initial solutions, a Variable Neighbourhood Search method and a Genetic Algorithm to improve the population, the latter two are interconnected. Computational experiments on benchmark data sets demonstrate that the proposed hybrid metaheuristic reaches high quality solutions in short computational times using fixed parameter settings.

Besides, a hybrid approach with an expert system and a genetic algorithm to production management in supply networks was also presented by Ławrynowicz [40, 41].

3.8 Planning and scheduling problems

Genetic algorithms have been also successfully implemented to solve various planning and scheduling problems. For example, Lee et al. [33] developed advanced planning and scheduling with outsourcing in manufacturing supply chains. The proposed model considers alternative processes plans for different job types.

Chen and Ji [11] proposed a genetic algorithm for dynamic advanced planning and scheduling with frozen interval. This paper investigates a dynamic advanced planning and scheduling (DAPS) problem where new orders arrive on a continuous basis. A periodic policy with a frozen interval is adopted to increase stability on the shop floor. A genetic algorithm is developed to find a schedule such that both production idle time and penalties on tardiness and earliness of both original orders and new orders are minimized at each rescheduling point. The numerical results confirm that the proposed methodology can improve the schedule stability while retaining efficiency.

A hybrid approach for control problems in a node of the supply network was published by Ławrynowicz [39]. This approach takes into account the loops in supply networks. In this approach, the production planning problem is first solved, and then the scheduling problem is considered within the constraints of the solution. The main objectives of this approach are to produce an Advanced Production Management (APRM) model that minimizes the makespan by considering alternative machines, alternative sequences of operations with precedence constrains, and outsourcing.

Fig. 7 shows the outline of the idea of planning and scheduling using an expert system and genetic algorithms. The first phase involves using a traditional approach combined with the genetic algorithm to produce a preliminary and possibly suboptimal schedule. The second phase uses a combination of an expert system and a genetic algorithm to construct a detailed schedule according to the detailed production plan.

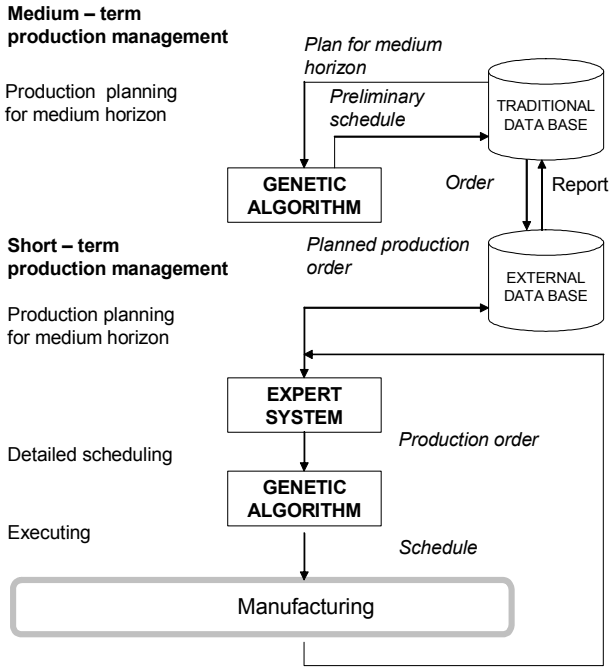

Figure 7. The outline of the idea of operations management using a genetic algorithm
(source: Ławrynowicz [39])

As shown in Fig. 7, proposed hybrid system does not only offer short-term production planning and scheduling to meet changing market requirements that can better utilize the available capacity of manufacturing systems, but also provides support for control. In this approach, the work-piece is one job. Each work-piece (i.e. job) has a unique priority indicator according to the order of the customer. The expert system creating detailed production plans takes into account the planned production orders and work-in process from the report. The report includes scheduled operations, which cannot be performed. In such situations, both kinds of orders – the parts of the production orders (from the report) and whole planned production orders – are an input to the expert system. The job requires different types of production resources. All resources are available in a limited capacity only. Detailed production planning matches future production load and capacities by generating detailed plans that determine the flow of materials and uses of resources over a given planning horizon. In the era of supply network, decisions on the use of resources should concern both internal and external capacities; the internal flow of materials should be synchronized with the incoming and outgoing flows. Therefore, the expert system generates detailed production plans based on available resources of the supply network. The expert system creates a detailed production plan as follows. The first step involves updating the planned production orders. In the second step, a human expert determines the top limit of priority indicator for orders. In the third step, a human expert selects m-th machine (bottleneck). Then the expert system automatically works out a sum of requirement capacity for m-th machine. After capacity requirement evaluation, the expert system compares the available capacity with capacity requirements. If the sum of requirement capacity is 70–100% of available capacity, then the expert system automatically creates a production plan from orders with a priority indicator smaller than or equal to the top limit indicator. In other cases, during an interactive dialogue a human expert makes a decision:

- is it possible to accept the sum of loads smaller than 70% of capacity of the machine?
- is it necessary to use an alternative processing plan or outsourcing?
- is division of lot-size possible?

The expert system will generate a production order according to the answers of the human expert. Next, the genetic algorithm with the operation-based encoding method is used. The proposed intelligent methods can be applied when there is a need to re-planning or re-scheduling. It is common knowledge that in a real-life factory there are often disruptions in production.

In such situations, the expert system and genetic algorithm executes re-planning and re-scheduling very quickly. In this experiment, the genetic algorithm was used the well-known roulette wheel selector and the next population was created using the partial match crossover (PMX) operator.

3.9 Multi-factory scheduling problem

Few researchers have considered methods with genetic algorithms to support scheduling in distribution manufacturing systems. Generally, distributed scheduling problems deal with the assignment of jobs to suitable factories and determine their production scheduling accordingly (Chan, et al. [7]). For example, Chan et al. [7] proposed an optimization algorithm named Genetic Algorithm with Dominated Genes (GADG) to solve distributed production scheduling problems with alternative production routings. In this approach, each chromosome represents a solution corresponding to:

(i) the allocation of jobs to factories,

(ii) the production priority of each job's operation in each machine in the network.

A chromosome is composed of genes. Each gene consists of five parameters (i.e. FMJOD), representing:

- Factory number (F),
- Machine number (M),
- Job number (J),
- Operation number (O) of the job, and
- Domination of the gene (D).

Fig. 8a shows a sample coding of a chromosome for the allocation and scheduling of three jobs to two factories, in which each factory has three machines, and each job requires three operations for completion. Assuming each operation requires one unit of production lead time, the scheduling result is shown in Fig. 8b.

In Fig. 8a, the first gene (11111) represents that O1 (Operation 1) of J1 (Job 1) is allocated to F1'sM1 (Factory 1's Machine 1), and it is a dominated gene denoted by 1. This coding can also be used to model alternative routings, for example the first gene can also be coded as (13111) to represent that O1 of J1 is allocated to F1's M3, i.e. the J1O1 can be operated inM1 orM3. The scheduling priority of jobs into machines is defined by the ordering, from the highest priority on the left to lowest on the right.

Therefore, Fig. 8a indicates that O1 of J1 (i.e. gene: 11111) will be scheduled before O1 of J3 (gene: 11311) in F1's M1. For each operation, if its preceding operation is not yet allocated, it will not be considered until the allocation of its preceding one is done. The scheduling will then move to consider the next gene, such as the second one (12330). This gene (12330) will only be allocated after its preceding operation (gene: 13320) has been allocated, as shown in Fig. 8b.

a) 11111-12330-12120-13130-22210-21220-23230-11311-13320

b)

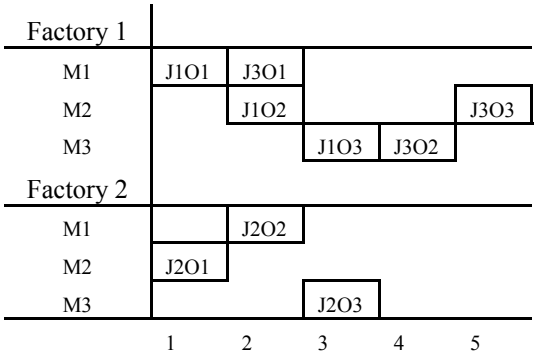

Figure 8. (a) A sample coding of chromosome (b) Scheduling result of sample chromosome *(source: Chan et al. [7])*

Dominated Gene (DG) indicates that this gene can increase the strength (fitness) of the chromosome. Initially, some genes in a new chromosome are randomly assigned to be dominated genes denoted by 1 in the D parameter of the chromosome, otherwise 0. Each chromosome may contain empty, 1, or more than 1 dominated gene. During evolutions, only those DGs undergo crossover in each pair of parents to generate a pair of offspring. Each offspring reserves most of the genes from one of the parents and inherits only the DGs from another parent. If these inherited DGs make the offspring stronger than the parent, they will remain dominated in the offspring, otherwise they will become normal genes. This idea is to identify and record the best genes, and ensure they will be passed to the offspring. GADG implements the idea of adaptive strategy. In this approach, a new crossover mechanism named dominated gene crossover has been introduced to enhance the performance of genetic search, and eliminate the problem of determining an optimal crossover rate. A number of experiments have been carried out. The results indicate that significant improvement could be obtained by the proposed algorithm.

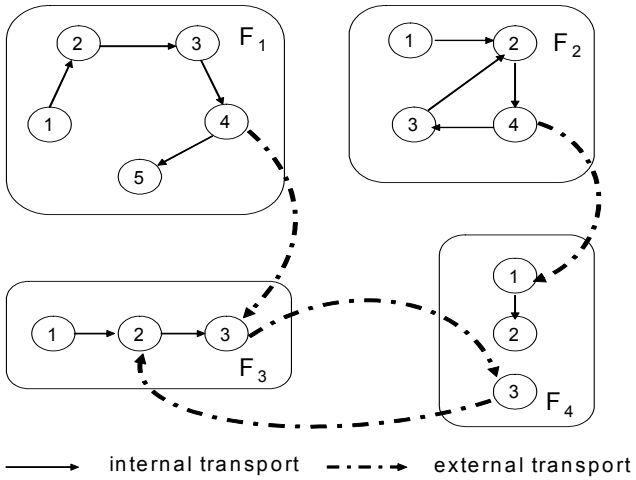

Figure 9. Relationships among jobs, resources, and factories
(source: own study)

Beside, an integration of the genetic algorithm and Gantt chart (GC) for job shop scheduling in distributed manufacturing systems has been also proposed by Jia et al. [28]. The integration of GA–GC is shown to be efficient at solving small-sized or medium-sized scheduling problems for a distributed manufacturing system. Multiple objectives can be achieved, including minimizing the makespan, job tardiness, or manufacturing cost.

Application of the genetic approach for advanced planning in multi-factory environment is also presented in the work of Chung et al. [16]. The proposed algorithm adopts the idea of dominant gene proposed by Chan et al. [7]. The model is subject to capacity constraints, precedence relationships, and alternative machines with different processing time. The objective function is to minimize the makespan, which consists of the processing time, the transportation time between resources either within the same factory or across two different factories, and the machine set-up time among operations. The results show the robustness of the proposed algorithm for this problem.

As shown above, despite many advantages in solving scheduling problems with genetic algorithms, the application of the above mentioned algorithms is questionable. Frequently, the loops in supply networks are not taken into consideration in many works.

4 A new approach to the scheduling problem in industrial clusters

In the industrial cluster, multiple factories can be selected to manufacture the products. The factories may be located in geographically distributed location, but situated near. In the literature, the term "industrial cluster" is widely used, it is defined as "a geographical and sectoral concentration and combination of firms" (Niu [46]). From the viewpoint of relationships, it is a local supply network based on partnership. The relationships between members within an industrial cluster are shown in Fig. 9.

In the industrial cluster, the individual operating decision making is dependent on the resources of the other factories, and the possibilities of the individual organization to utilize these resources are determined by their place in the network. In many cases, the industrial cluster is a distributed manufacturing system.

In the research of Ławrynowicz [37], a typical industrial cluster, which has J different tasks (products) (1, 2, ..., m) for F factories (1, 2, …, r) is considered. Each factory has R resources (1, 2, ..., q). All jobs are loaded, according to the predetermined technological sequence given in processing plans.

The routes for the jobs are such that a job may visit some resources and use some transportation more than once. There are several constraints on jobs and resources:

1) there are no precedence constraints among operations of different jobs;
2) operations cannot be interrupted and each resource can handle only one job at a time;
3) each job can be performed only on one resource at a time.

In this approach, the processing plans of jobs can include also external transport operations. The objective is to minimize the total makespan of the industrial cluster.

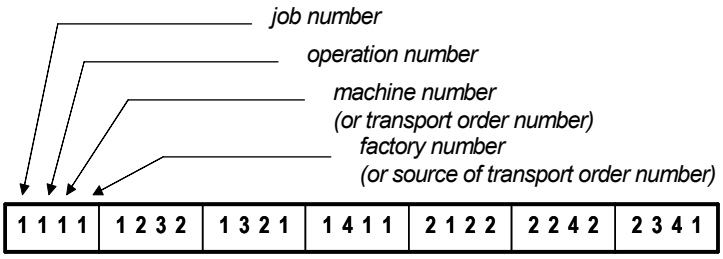

Figure 10. Example of a chromosome type A
(soure: own study)

The following notation is used for optimization of scheduling in the industrial cluster (Ławrynowicz [37]):

m - number of jobs,

p - number of operations,

q - number of resources,

r - number of factories,

J_j - the j-th job, where j = 1, …, m,

O_i - the i-th operation, where i= 1, …, p,

R_n - the n-th resource where n = 1, …, q,

F_k - the k-th factory, where k = 1, …, r,

P_o - the o-th transport order, where o=1, …, q-2 and o>2,

S_t - the t-th source of transport order o, where t=r+1, …, r+m,

T_{ji} - the time of operation i of job j.

In this approach, the source of the transport order is the job. If a considered system includes three factories then the sources of transport orders are denoted as follows: for the first job the source of transport orders is denoted by S_{3+1} i.e. S_4, for the second job the source of transport orders is denoted by S_5, for the third job the source of transport orders is denoted by S_6 etc.

From the mathematical point of view, an industrial cluster is a digraph, which has loops and therefore the methods based on "network theory" cannot be easily adopted in supply network management. When the job shop problem is not too large, the methods proposed in the literature are able to obtain the optimal solution within reasonable time. But its implementation would not be easy with conventional information systems.

Therefore, the author proposes a new approach to the distributed scheduling in the industrial cluster which uses a modified genetic algorithm (MGA). The modified genetic algorithm proposed by the author creates schedule for each factory and enables transport order planning. The MGA is an improved version of prototypes developed by the authors in early stages of this research (Ławrynowicz [39, 40 and 41].

The design of a suitable chromosome is the first step for a successful genetic algorithm implementation because it applies probabilistic transition rule on each chromosome to create a population of chromosomes, representing a good candidate solution.

Particularly, in the industrial cluster where jobs will be dispatched to many factories, the encoding of the scheduling problems plays an important role to implement effective operations management methods. As mentioned above, in the scheduling problem, the popular encoding is operation-based method. This representation encodes a schedule as a sequence of operations and each gene stands for one operation. By this idea, the author creates new encoding method for a scheduling problem in the industrial cluster. In this approach, a modified genetic algorithm employs two steps to encode the scheduling problem. According to the step, two different types of chromosomes are designed.

In the first step, each chromosome type A represents a potential optimal solution of a problem being optimized. Chromosome type A consists of a set of 4-positions gene. The chromosome structure can be represented as shown in Fig. 10, where the value of the first position of the gene represents the job, the value of the second position the operation number, and the next two values the pair as follows: the resource number and the factory number or the transport order number and the source of the transport order number.

The second step is to copy the first and the second position from the gene of the chromosome A into the gene of the chromosome B, and to translate the last two positions from the gene of the chromosome A into one position gene of the chromosome B. Chromosome type B is designed, as follows.

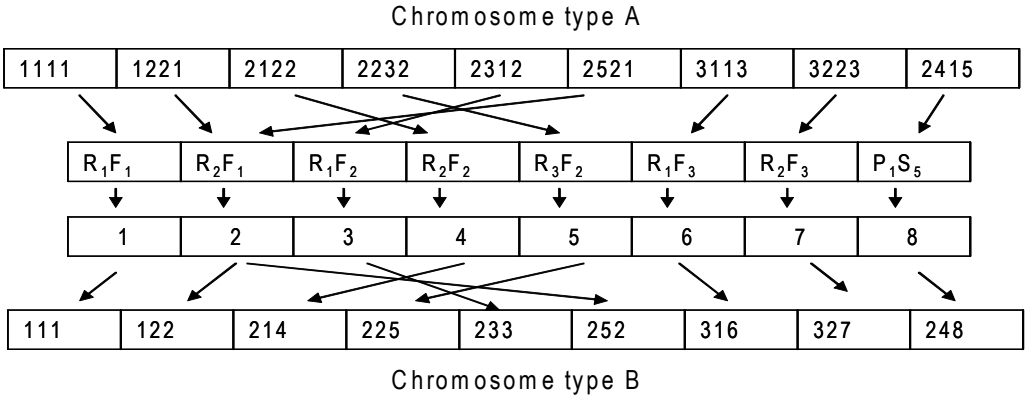

Figure 11. Example of translation
(source: own study)

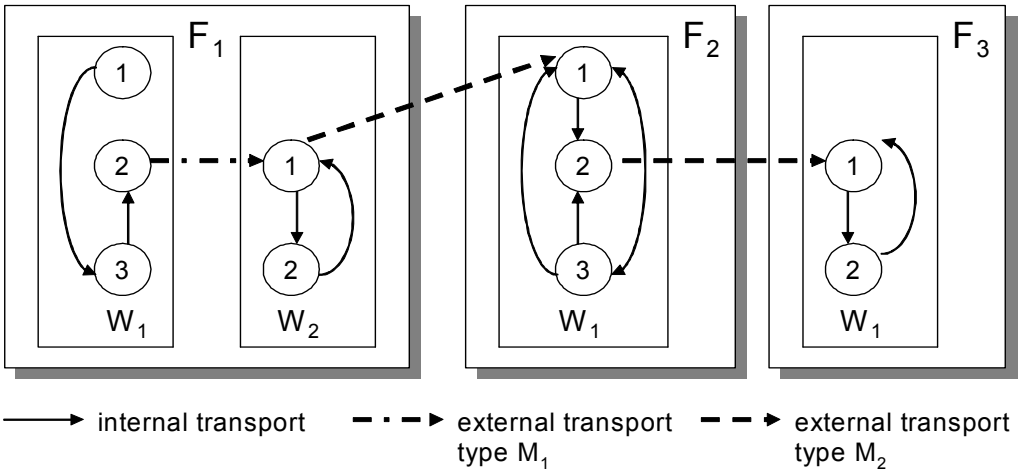

Figure 12. Relationships among jobs, resources, workshops and factories
(source: own study)

Similarly as chromosome type A, the first position represents the job, and the second the operation number, but the last position contains a unique number of the resource.

Fig. 11 shows the way of a translation. Thus, the new encoding method includes both manufacture operations and long transport operations. In procedure of this MGA, two new steps are added (to CGA). The first step is added in the beginning and consists of a translation of the chromosome type A into the chromosome type B. Thus, the initial population is created for type B chromosome.

The other operation is added after the determination of the best chromosome of type B (which gives the smallest value of the makespan using the genetic algorithm with classical encoding method) and consists of a translation of the best chromosome of type B into type A chromosome. In the MGA, the most popular selection method that is referred to as roulette wheel selector was used.

The next population was created using the partial mapped crossover operator (PMX), and the mutation was a random interchange of values in two positions. The number of generations was used as a stopping measure. In the work by Ławrynowicz [37, 38], representative examples are provided to show that the above suggested method can improve distributed scheduling in industrial clusters.

Beside, the author proposed a new genetic algorithm for a distributed scheduling in a supply network (Ławrynowicz [36]) where each chromosome is a set of 5-position genes. The new genetic algorithm enables not only a manufacturing scheduling in supply networks. Additionally, the new genetic algorithm aided planners in transport orders planning.

Fig. 12 shows an example of the relationships among the jobs, resources and factories for a production plan of supply network which was considered by the author. Basing on the above idea of operation codes with 4-position genes, the author developed the new genetic algorithm, where each chromosome is a set of 5-position genes. In proposed by the author encoding method, the value of the first position of the gene represents the job, the value of the second position the operation number, and the next three values the segment X or Y as follows (accordingly): the resource number, the workshop number and the factory number or the transport type number, the transport order number and the source of the transport order number. The results of the experiments show that the proposed new genetic algorithm is a very efficient and effective algorithm.

5 Conclusion

This paper describes how the genetic algorithms have been applied to the optimization of manufacturing scheduling problems. Representation scheme of a feasible solution to the considered problem is a key aspect of evolutionary algorithms. Therefore, in this study, the focus is brought on the coding problems.

It is common knowledge that in solving large-size problems, genetic algorithms show much better performance (Chung et al. [16]). Despite many advantages in solving scheduling problems presented in the existing literature, many applications of genetic algorithms are questionable. As mentioned above, researchers still study small-scale problems or only flow shop problems, where there are many constraints. It is possible that equally important and stimulating research unknown to the authors was unintentionally omitted.

Many genetic algorithms proposed in the literature have been created for scheduling in a single factory. The approach often ignores dividing jobs and interactions between the various firms within supply networks at operations management level in order to improve manufacturing processes. But, in the era of supply network, decisions on the use of resources should concern both internal and external capacities; the internal flow of materials should be synchronized with the incoming and outgoing flows. For this purpose, a system for scheduling must take into consideration the possibility of dividing jobs into factories, loops, and a long transport. Therefore, the author proposes modified genetic algorithm (MGA), which take into account loops in supply networks. Additionally, the proposed modified genetic algorithm enables dividing jobs between factories, and transport orders planning in the industrial cluster.

Summarizing, advances in genetic algorithms create new prospects for inter-organizational cooperation.

As mentioned above, the main objective of this paper is to present heuristic methods based on genetic algorithms. But, it is noted that another group of researchers proposed an ant colony optimization (ACO) for solving advanced scheduling problem (Rajendran and Ziegler [50], Panahi et al. [48]). Ant algorithms are optimization algorithms inspired by the foraging behaviour of real ants in the wild (Mullen et al. [43]). Within the Artificial Intelligence (AI) community, ant algorithms are considered under the category of swarm intelligence. Swarm intelligence encompasses the implementation of intelligent multi-agent systems that are based on the behaviour of real world insect swarms, as a problem solving tool. Future research can also investigate the possibility of incorporating the proposed ACO for solving scheduling problems in the industry.

6 References

[1] Allahverdi A., Ng C.T., Cheng T.C.E., Kovalyov M.Y. - *A survey of scheduling problems with setup times or costs* [in] European Journal of Operational Research, Vol. 187, 2008, pp. 985-1032.

[2] Arroyo J.E.C., Armentano V.A. - *Genetic local search for multi-objective flow shop scheduling problems* [in] European Journal of Operational Research, Vol. 167, 2005, pp. 717-738.

[3] Balin S. - *Non-identical parallel machine scheduling using genetic algorithm* [in] Expert Systems with Applications, Vol. 38, 2011, pp. 6814-6821.

[4] Blanco A., Delgado M., Pegalajar M.C. - *A real-coded genetic algorithm for training recurrent neural networks* [in] Neural Networks, Vol. 14, 2001, pp. 93-105.

[5] Braglia M., Grassi A. - *A new heuristic for the flow-shop scheduling problem to minimize makespan and maximum tardiness* [in] International Journal of Production Research, Vol. 47, No. 1, 2009, pp. 273-288.

[6] Carlos A., Coello C. - *Theoretical and numerical constraint-handling techniques used with evolutionary algorithms: a survey of the state of the art* [in] Computer Methods in Applied Mechanics Engineering, Vol. 191, 2002, pp. 1245-1287.

[7] Chan F.T.S., Chung S.H., Chan P.L.Y. - *An adaptive genetic algorithm with dominated genes for distributed scheduling problems* [in] Expert System with Applications, Vol. 29, 2005, pp. 364-371.

[8] Chan F.T.S., Chung S.H., Chan L.Y. - *An introduction of dominant genes in genetic algorithm for FMS* [in] International Journal of Production Research, Vol. 46, No. 16, 2008, pp. 4369-4389.

[9] Chang W.D. - *Nonlinear system identification and control using a real-coded genetic algorithm* [in] Applied Mathematical Modelling, Vol. 31, 2007, pp. 541-550.

[10] Chang P.C., Chen S.H., Lin K.L. - *Two-phase sub population genetic algorithm for parallel machine-scheduling problem* [in] Expert Systems with Applications, Vol. 29, 2005, pp. 705-712.

[11] Chen K.J., Ji P. - *A genetic algorithm for dynamic advanced planning and scheduling (DAPS) with frozen interval* [in] Expert Systems with Applications, Vol. 33, 2007, pp. 1004-1010.

[12] Chen J.S., Pan J.C.H., Lin C.M. - *A hybrid genetic algorithm for the re-entrant flow-shop scheduling problem* [in] Expert Systems with Applications, Vol. 34, Chiang 2008, pp. 570-577.

[13] Cheng R., Gen M., Tsujimura Y. - *A tutorial survey of job-shop scheduling problems using genetic algorithms. Part 1. Representation* [in] Computers and Industrial Engineering, Vol. 30, No. 4, 1996, pp. 983-997.

[14] Cheng R., Gen M., Tsujimura Y. - *A tutorial survey of job-shop scheduling problems using genetic algorithms. Part II: Hybrid genetic search strategies* [in] Computers and Industrial Engineering, Vol. 36, 1999, pp. 343-364.

[15] Chiang T.C., Cheng H.C., Fu L.C. - *NNMA: An effective memetic algorithm for solving multiobjective permutation flow shop scheduling problems* [in] Expert Systems with Applications, Vol. 38, 2011, pp. 5986-5999.

[16] Chung S.H., Lau H.C.W., Choy K.L., Ho G.T.S., Tse Y.K. - *Application of genetic approach for advanced planning in multi-factory environment* [in] International Journal of Production Economics, Vol. 127, 2010, pp. 300-308.

[17] Davis L. - *Applying adaptive algorithms to epistatic domains* [at] The International Joint Conference on Artificial Intelligence, 1985, pp. 162-164.

[18] França P.M., Gupta J.N.D., Mendes A.S., Moscato P., Veltink K.J. - *Evolutionary algorithms for scheduling a flowshop manufacturing cell with sequence dependent family setups* [in] Computers & Industrial Engineering, Vol. 48, 2005, pp. 491-506.

[19] Gao J., Gen M., Sun L., Zhao X. - *A hybrid of genetic algorithm and bottleneck shifting for multi-objective flexible job shop scheduling problems* [in] Computers & Industrial Engineering, Vol. 53, 2007, pp. 149-162.

[20] Gao J., He G., Wang Y. - *A new parallel genetic algorithm for solving multiobjective scheduling problems subjected to special process constraint* [in] The International Journal of Advanced Manufacturing Technology, Vol. 43, 2009, pp.151-160.

[21] Gao, J., Sun L., Gen M. - *A hybrid genetic and variable neighborhood descent algorithm for flexible job shop scheduling problems* [in] Computers & Operations Research, Vol. 35, No. 9, 2008, pp. 2892-2907.

[22] Gao L., Zhang G., Zhang L., Li X. - *An efficient memetic algorithm for solving the job shop scheduling problem* [in] Computers & Industrial Engineering, Vol. 60, 2011, pp. 699-705.

[23] Gholami M., Zandieh M. - *Integrating simulation and genetic algorithm to schedule a dynamic flexible job shop* [in] Journal of Intelligent Manufacturing, Vol. 20, 2009, pp. 481-498.

[24] Goldberg D., Lingle R. - *Alleles, loci and the traveling salesman problem* [at] The First International Conference on Genetic Algorithms, Hillsdale 1985, pp. 154-159.

[25] Goldberg D.E. - *Genetic algorithms in search, optimization and machine learning*. Addison-Wesley, Reading, MA, 1989.

[26] Guang Y., Hong Z.W. - *Optimization of tool change timing in a nut forming process using genetic algorithms* [in] Journal of Intelligent Manufacturing, Vol. 15, 2004, pp. 693-699.

[27] Jarboui B., Ibrahim S., Siarry P., Rebai A. - *A combinatorial particle swarm optimization for solving permutation flowshop problems* [in] Computers & Industrial Engineering, Vol. 54, 2008, pp. 526-538.

[28] Jia H.Z., Fuh J.Y.H., Nee A.Y.C., Zhang Y.F. - *Integration of genetic algorithm and Gantt chart for job shop scheduling in distributed manufacturing systems* [in] Computers & Industrial Engineering, Vol. 53, 2007, pp. 313-320.

[29] Jolai F., Amalnick M.S., Alinaghian M., Shakhsi-Niaei M., Omrani H. - *A hybrid memetic algorithm for maximizing the weighted number of just-in-time jobs on unrelated parallel machines* [in] Journal of Intelligent Manufacturing, Vol. 22, 2011, pp. 247-261.

[30] Kim K., Jeong I.J. - *Flow shop scheduling with no-wait flexible lot streaming using an adaptive genetic algorithm* [in] The International Journal of Advanced Manufacturing Technology, Vol. 44, 2009, pp. 1181-1190.

[31] Kobbacy K.A.H., Vadera S., Rasmy M.H. - *AI and OR in management of operations: history and trends* [in] Journal of the Operational Research Society, Vol. 58, No. 1, 2007, pp. 10-28.

[32] Liaw C.F. - *A hybrid genetic algorithm for the open shop scheduling problem* [in] European Journal of Operational Research, Vol. 124, 2000, pp. 28-42.

[33] Lee Y.H., Jeong Ch. S., Moon Ch. *Advanced planning and scheduling with outsourcing in manufacturing supply chain* [in] Computers & Industrial Engineering, Vol. 43, 2002, pp. 351-374.

[34] Liao L.M., Tsai C.H. - *Heuristic algorithms for two-machine flow shop with availability constraints* [in] Computers & Industrial Engineering, Vol. 56, 2009, pp. 306-311.

[35] Low C., Yeh Y. - *Genetic algorithm-based heuristics for an open shop scheduling problem with set-up, processing, and removal times separated* [in] Robotics and Computer-Integrated Manufacturing, Vol. 25, 2009, pp. 314-322.

[36] Ławrynowicz A. - *A genetic algorithm for distributed scheduling in supply networks* [at] The 2nd Conference on Applied Operational Research - ICAOR'10, Turku, Finland. Lecture Notes in Management Science, Vol. 2, 2010, pp. 282-294.

[37] Ławrynowicz A. - *A new genetic algorithm for job shop scheduling in supply networks* [at] The Fourth European Conference on Intelligent Management Systems in Operations, Greater Manchester, 2009, pp. 101-110.

[38] Ławrynowicz A. - *A novel intelligent method for task scheduling in industrial cluster* [in] Advanced Information Technologies for Management - AITM 2009, Research Papers, No. 85, 2009, pp. 170-178.

[39] Ławrynowicz A. - *Integration of production planning and scheduling using an export system and a genetic algorithm* [in] Journal of the Operational Research Society, Vol. 59, No. 4, 2008, pp. 455-463.

[40] Ławrynowicz A. - *Hybrid approach with an expert system and a genetic algorithm to production management in the supply net* [in] Intelligent Systems in Accounting, Finance and Management, Vol. 14, No. 1-2, 2006, pp. 59-76.

[41] Ławrynowicz A. - *Production planning and control with outsourcing using artificial intelligence* [in] International Journal Services and Operations Management, Vol. 3, No. 2, 2007, pp. 193-209.

[42] Moon I., Lee S., Bae H. - *Genetic algorithms for job shop scheduling problems with alternative routings* [in] International Journal of Production Research, Vol. 10, 2008, pp. 2695-2705.

[43] Mullen R.J., Monekosso D., Barman S., Remagnino P. - *A review of ant algorithms* [in] Expert Systems with Applications, Vol. 36, 2009, pp. 9608-9617.

[44] Nagano M.S., Ruiz R., Lorena L.A.N. - *A Constructive Genetic Algorithm for permutation flowshop scheduling* [in] Computers & Industrial Engineering, Vol. 55, 2008, pp. 195-207.

[45] Nearchou A.C. - *The effect of various operators on the genetic search for large scheduling problems* [in] International Journal of Production Economics, Vol. 88, 2004, pp. 191-203.

[46] Niu K.H. - *The involvement of firms in industrial clusters: A conceptual analysis* [in] International Journal of Management, Vol. 26, No. 3, 2009, pp. 445-455.

[47] Onwubolu G., Davendra D. - *Scheduling flow shops using differential evolution algorithm* [in] European Journal of Operational Research, Vol. 171, 2006, pp. 674-692.

[48] Panahi H., Tavakkoli-Moghaddam R. - *Solving a multi-objective open shop scheduling problem by a novel hybrid ant colony optimization* [in] Expert Systems with Applications, Vol. 38, 2011, pp. 2817-2822.

[49] Prakash A., Chan F.T.S., Deshmukh S.G. - *FMS scheduling with knowledge based genetic algorithm approach* [in] Expert Systems with Applications, Vol. 38, 2011, pp. 3161-3171.

[50] Rajendran C., Ziegler H. - *Ant-colony algorithms for permutation flow shop scheduling to minimize makespan/total flowtime of jobs* [in] European Journal of Operational Research, Vol. 155, 2004, pp. 426-438.

[51] Rajkumar R., Shahabudeen P. - *An improved genetic algorithm for the flowshop scheduling problem* [in] International Journal of Production Research, Vol. 47, No. 1, 2009, pp. 233-249.

[52] Ruiz R., Maroto C. - *A genetic algorithm for hybrid flowshops with sequence dependent setup times and machine eligibility* [in] European Journal of Operational Research, Vol. 169, 2006, pp. 781-800.

[53] Ruiz R., Maroto C., Alcaraz J. - *Two new robust genetic algorithms for the flowshop scheduling problem* [in] Omega, Vol. 34, 2006, pp.461-476.

[54] Tavakkoli-Moghaddam R., Azarkish M., Sadeghnejad-Barkousaraie A. - *Solving a multi-objective job shop scheduling problem with sequence-dependent setup times by a Pareto archive PSO combined with genetic operators and VNS* [in] The International Journal of Advanced Manufacturing Technology, Vol. 53, 2011, pp. 733-750.

[55] Tseng L.Y., Lin Y.T. - *A genetic local search algorithm for minimizing total flowtime in the permutation flowshop scheduling problem* [in] International Journal of Production Economics, Vol. 127, 2010, pp. 121-128.

[56] Wang Y.M., Yin H.L., Wang J. - *Genetic algorithm with new encoding scheme for job shop scheduling* [in] The International Journal of Advanced Manufacturing Technology, Vol. 44, 2009, pp. 977-984.

[57] Vallada E., Ruiz R. - *A genetic algorithm for the unrelated parallel machine scheduling problem with sequence dependent setup limes* [in] European Journal of Operational Research, Vol. 211, 2011, pp. 612-622.

[58] Xu X., Xu Z., Gu X. - *An asynchronous genetic local search algorithm for the permutation flowshop scheduling problem with total flowtime minimization* [in] Expert Systems with Applications, Vol. 38, 2011, pp. 7970-7979.

[59] Ying-Hua C., Young-Chang H. - *Dynamic programming decision path encoding of genetic algorithms for production allocation problems* [in] Computers & Industrial Engineering, Vol. 54, 2008, pp. 53-65.

[60] Zhang G., Gao L., Shi Y. - *An effective genetic algorithm for the flexible job-shop scheduling problem* [in] Expert Systems with Applications, Vol. 38, 2011, pp. 3563-3573.

[61] Zhang C., Rao Y., Li P. - *An effective hybrid genetic algorithm for the job shop scheduling problem* [in] The International Journal of Advanced Manufacturing Technology, Vol. 39, 2008, pp. 965-974.

[62] Zhang R., Wu C. A. - *A hybrid approach to large-scale job shop scheduling* [in] Applied Intelligence, Vol. 32, 2010, pp. 47-59.

[63] Zegordi S.H., Abadi I.N.K., Nia M.A.B. - *A novel genetic algorithm for solving production and transportation scheduling in a two-stage supply chain* [in] Computers & Industrial Engineering, Vol. 58, 2010, pp. 373-381.

[64] Zobolas G.I., Tarantilis C.D., Ioannou G. - *A hybrid evolutionary algorithm for the job shop scheduling problem* [in] Journal of the Operational Research Society, Vol. 60, No. 2, 2009, pp. 221-235