Recent Developments in Evolutionary Computation for Manufacturing Optimization: Problems, Solutions, and Comparisons

Christos Dimopoulos and Ali M. S. Zalzala, *Member, IEEE*

***Abstract*—The use of intelligent techniques in the manufacturing field has been growing the last decades due to the fact that most manufacturing optimization problems are combinatorial and NP hard. This paper examines recent developments in the field of evolutionary computation for manufacturing optimization. Significant papers in various areas are highlighted, and comparisons of results are given wherever data are available. A wide range of problems is covered, from job shop and flow shop scheduling, to process planning and assembly line balancing.**

***Index Terms*—Assembly lines, cellular manufacturing, design, evolutionary computation, genetic algorithms, manufacturing optimization, process planning, scheduling.**

I. Introduction

SINCE the 1950's, some authors have been using concepts based on Darwin's evolution theory for the solution of optimization problems [1]–[3]. Numerous algorithms based on the same concepts have been developed over the last 30 years. They are usually described by the term "evolutionary computation methods." The most notable members of this group are simple genetic algorithms (GA's) [4], [5], evolution strategies [6], evolutionary programming [7], classifier systems [8], and genetic programming [9]. Bäck *et al.* [10] give an excellent review of evolutionary computation methods, and highlight some recent developments in the field. For the reader not familiar with evolutionary computation concepts, additional information can be found in [11] and [12].

A large number of combinatorial problems are associated with manufacturing optimization. Most of them are NP complete, i.e., there is no polynomial-time algorithm that can possibly solve them, unless it is proved that $P = \text{NP}$ [13]. Heuristic methods are normally employed for the solution of these problems. A growing number of researchers have adopted the use of meta-heuristic techniques ("smart heuristics") for large combinatorial problems. Evolutionary computation methods are meta heuristics that are able to search large regions of the solution's space without being trapped in local optima. Some other well-known meta heuristics are simulated annealing (SA) [14] and tabu search [15]–[17].

The aim of this paper is to illustrate recent developments in the field of evolutionary computation for manufacturing optimization. A wide range of optimization problems is considered, from the classic job-shop and flow-shop scheduling problems, to assembly line balancing and aggregate production planning. We focus mainly on recent publications, but there are pointers to significant earlier approaches. In this way, the reader who is interested in a particular problem can use this paper as a starting point. The term "evolutionary algorithms" (EA's) is used interchangeably in this paper to describe different evolutionary computation methods.

The rest of the paper is organized as follows. Section II examines recent evolutionary algorithms for the job-shop scheduling problem. The same procedure is followed in Section III for the flow-shop scheduling problem, in Section IV for the dynamic scheduling problem, in Section V for the process planning problem, in Section VI for cellular manufacturing optimization problems, in Section VII for assembly optimization problems, and in Section VIII for design optimization problems. Section IX overviews some recent developments in other manufacturing optimization areas, and Section X draws the conclusions of this paper.

II. The Job-Shop Scheduling Problem

A. Introduction and Historical Development

Considerable work in the field of evolutionary computation has been devoted to the solution of the job-shop scheduling problem (JSSP). Davis [18] made the first attempt to solve the problem more than ten years ago using the concept of preference lists, which will be explained later in this section. Yamada and Nakano [20] later proposed a more natural representation for the solution of the problem using the completion times of operations. Since then, the number of relevant publications has been growing rapidly, and so has the number of different approaches that have been proposed for solving the problem.

B. Formulation of the Problem

The job-shop scheduling problem consists of ordering n jobs to be processed in m machines. Each job involves a number of different machining operations. The following conditions hold for the classic formulation of the JSSP:

- each machine can process only one job at a time
- the sequence of operations for each job is predefined

Manuscript received June 1, 1998; revised December 4, 1998 and March 1, 1999. The work of C. Dimopoulos was supported by the Greek State Fund (I.K.Y.).

C. Dimopoulos is with the Department of Automatic Control and Systems Engineering, University of Sheffield, Sheffield S1 3JD, U.K. (e-mail: cop97cd@sheffield.ac.uk).

A. M. S. Zalzala is with the Department of Computing and Electrical Engineering, Heriot-Watt University, Edinburgh, EH14 4AS, Scotland, U.K. (e-mail: a.zalzala@hw.ac.uk).

Publisher Item Identifier S 1089-778X(00)02628-X.

- two operations of the same job cannot be processed at the same time
- preemption is not allowed (an operation cannot be withdrawn from a machine unless it is completed)
- processing times are known in advance
- transportation time between machines is zero.

The quality criterion most often used for the JSSP is the minimization of makespan $(C_{\max})$. Makespan is defined as the completion time of the final job to leave the system [38]. The JSSP is also known as the $n/m/G/C_{\max}$ [21] problem. Bierwirth *et al.* [22] describe it as a "representative of constrained combinatorial problems." Garey *et al.* [23] have illustrated that it is NP hard in the strong sense (proof by transformation of the 3-PARTITION problem to the associated JSSP decision problem). In this section, we consider the static version of the problem, in which unexpected events are not taken in account. The dynamic version of the JSSP will be discussed in a following section. In the special case of $n = m$, the problem is described as the "generalized assignment problem." An EA-based heuristic for the solution of this problem has been proposed by Chu and Beasley [24].

1) Variations from the Basic Form: Cao *et al.* [26] argue that the classic formulation of the JSSP is unrealistic since it does not take into account a number of elements which are important in real-life scheduling, like setup times, due dates, and machine off-line times. Academic research has been criticized for considering scheduling problems that rarely appear in practice [27], [28]. As a result, many researchers in the field of evolutionary computation are now using a variety of criteria for the evaluation of schedules. Minimization of makespan is still used as an objective in many cases [22], [29]–[31], but the general belief is that the objective of manufacturing optimization should be the minimization of production cost. Addressing the overdominance of makespan-oriented work in the field, Fang *et al.* [32] employed seven quality criteria for the evaluation of good schedules: maximum tardiness, average tardiness, weighted flow time, weighted lateness, weighted tardiness, weighted number of tardy jobs, and weighted earliness plus weighted tardiness. The last criterion is in accordance with the just-in-time (JIT) principle of having a product made exactly when it is required. This minimizes storage costs (earliness) and lateness fines (tardiness). Similar objectives are used in [33], [19], [35]–[37]. Due dates and ready times of the products are prespecified in these cases. An excellent overview of scheduling objectives can be found in [21] and [38].

C. Encoding

The classic binary solution representation of the simple genetic algorithm has rarely been used for the JSSP. Purpose-based representations can be much more effective. In the following paragraphs, we will attempt a classification of the most successful representations.

1) Direct Representations: Perhaps the most natural representation for the solution of the problem would be a data structure that can be used as a schedule itself. No decoding is needed to obtain the schedule; thus, this type of representation is called direct. (Note that we use a different classification method from Cheng *et al.* [43].) A data structure comprising a "direct" representation according to our classification needs no transformation at all. Burns [39] was perhaps the first researcher to employ an EA with direct representation for the solution of a production scheduling problem. His representation explicitly defined the process plan for each job, machine assignment for each operation, and individual start–end times. A traditional scheduling algorithm initialized the population with feasible solutions, and purpose-based operators ensured that solutions remained valid throughout the evolutionary procedure (a discussion on variation operators will follow in the section on flow-shop scheduling). An alternative approach is the use of an m-partitioned permutation (where m is the total number of machines), with each partition representing the complete schedule of an independent machine. This representation is especially popular in sequencing problems, where the solution is not partitioned, so all well-known traveling salesman problem (TSP) operators can be easily applied. Dagli and Sittisathancai [37], [40] employed this type of direct representation for the solution of the JSSP. They overcame feasibility problems by using legal schedules to initialize the population and an order-based crossover operator to preserve the precedence constraints of the problem. They also used a back-propagation neural network for the evaluation of schedules. Aizpuru and Usunariz [41] adopted the same representation for their hybrid scheduling algorithm, which was based on evolutionary algorithms and tabu search. A knowledge-based system was employed to generate efficient scheduling strategies. The hybrid algorithm helped the system to induce knowledge about the scheduling procedure. Giffler and Thompson's (GT) [42] algorithm generated initial actives schedules, and efficient operators maintained the precedence relations of the jobs.

2) Indirect Representations:

a) Job-Based Representations: The common type of indirect representation does not explicitly state the operation number, but instead, only the owing job is defined. The chromosome

$$[J_1, J_2, J_1, J_3, J_2, J_1, \cdots]$$

indicates that the first operation of the first job should be scheduled first, followed by the first operation of the second job, the second operation of the first job, etc. It is obvious that a schedule builder is needed to transform this solution into a feasible schedule (for discussions about schedule builders, see [43]–[45]). Bierwirth *et al.* [22] employed this method in their discussion of permutation representations for combinatorial problems. Their experimentation with various crossover operators led to the conclusion that the preservation of the absolute order of jobs and their associated operations was quite significant for the JSSP. They introduced a new operator called PPX (precedence preservation operator) that featured this useful characteristic. Fang *et al.* [32] highlighted the superiority of a job-based GA over dispatching rules and stochastic hill climbing on a variety of scheduling criteria. Shi [46], [47] built an EA scheduler that efficiently decoded the strings into active schedules, and utilized operators optimized for speed.

b) Dispatching Rule Representations: The use of dispatching rules for scheduling is a common manufacturing practice [49]–[51]. Blackstone *et al.* [50] give the following definition: "A dispatching rule is used to select the next job to be processed from a set of jobs awaiting service." In the case of static scheduling, this selection is based on various job characteristics, such as processing time, due date, etc.

Herrmann *et al.* [48] proposed an efficient EA representation, which was based on dispatching rules. The solution was encoded in the following form:

$$[\text{EDD}, \text{SPT}, \text{FIFO}, \cdots]$$

where EDD is the earliest due date rule, SPT is the shortest processing time rule, and FIFO is first in, first out rule.

Each element represented a machine, and the value of the element defined the dispatching rule that this machine used for the scheduling of waiting operations. This type of representation did not suffer from feasibility problems, and the application of operators was straightforward. Fujimoto *et al.* [52], [53] employed the same representation for the scheduling of a flexible manufacturing system (FMS, a computer-controlled grouping of semi-independent workstations, linked by automated material-handling systems). Each element corresponded to a decision-making point in the plant, and the value of the element specified the dispatching rule that would be used at this point. Kumar and Srinivasan [36] used a circular string of dispatching rules as a scheduling policy, whenever a part was requested for processing.

Dorndorf and Pesch [30] proposed an alternative use of the same representation for the JSSP, where each rule determined the next job to be scheduled among the conflict set of jobs created by Giffler and Thompson's algorithm. However, this method performed poorly in comparison with another algorithm presented in the same paper based on the shifting bottleneck heuristic, a well-known method for the solution of the JSSP. An EA controlled the selection of nodes in the enumeration tree created by the heuristic.

Finally, Fang *et al.* [25] employed the dispatching rules representation for the solution of an open-shop scheduling problem (the case where the sequence of operations is not predefined).

c) Preference-List Representations: A popular way of encoding a solution of the JSSP is the preference-list representation. Preference lists are not actual schedules, but a preferable sequence of operations on each machine. Operations are scheduled according to this sequence unless they violate a precedence constraint. In that case, the next operation in the preference list is scheduled. Croce *et al.* [33] used the concept of preference lists for the encoding of solutions, together with a look-ahead evaluation method which generated nondelay schedules (see Baker [57] for a discussion on schedule types). Cao *et al.* [26] addressed a complex JSSP problem with multiple objectives utilizing a hierarchical evaluation (HE) model instead of a look-ahead evaluation. Their framework was able to generate feasible schedules and perform local optimization at the same time, resulting in slightly better performance than Croce's algorithm. Kobayashi *et al.* [58] and Ono *et al.* [59] encoded the solution in the same preference-list form. They additionally introduced two purpose-based crossover operators: the subsequent exchange crossover (SSX) and job-order based crossover (JOX), respectively. JOX used the traditional Giefler and Thompson (GT) algorithm for the decoding of solutions into active schedules. The result was a much better performance, both in terms of optimal and average values for Fisher and Thompson's benchmark problems (see later paragraph). Park and Park [60], [61] reported their preference list-based GA with the introduction of a crossover operator, called the active schedule constructive crossover (ASCX), which was based on the active schedule generation algorithm [57].

d) Alternative Representations: Several other representation schemes have been reported. Perhaps the most successful was proposed by Kim and Lee [62], [63]. Their schedule representation was basically a priority list of operation–machine assignment pairs, which corresponded to a certain priority rule. Schedules (and consequently, the corresponding priority rules) were refined evolutionarily with the help of a genetic reinforcement learning (GRL) procedure. While no computational times were reported, this method showed the best overall performance on Muth and Thompson's benchmark problems in comparison with every other evolutionary method included in this survey.

Yamada and Nakano [64] employed a disjunctive-graph representation for the solution of the JSSP. Following the trend of enhancing the evolutionary process with local search techniques, they introduced a crossover operator called multistep crossover (MSX), which was, in effect, a local search operator. Cho *et al.* [65] presented a method called the total operation order method (TOOM) where a solution was given in the form of a job operation matrix, which defined the absolute order of all operations to be processed. Liang and Mannion [66] proposed a sparse matrix solution's representation with purpose-based operators to ensure the feasibility of solutions. A dynamic data structure called the "hierarchical linked list" was utilized by Niemeyer and Shiroma [54] in order to accommodate variable lengths of jobs and operations in a real manufacturing environment. Kim and Kim [55] tackled the problem of infeasibility by using a random-keys [56] representation for the solutions. Finally, Gohtoh *et al.* [68] applied a special EA with neutral mutations [69] to some standard benchmark problems. An excellent analytical review of the representations that have been used for the solution of the JSSP can be found in [43].

D. Test Problems and Case Studies

Evolutionary computation methods have not been adopted in standard manufacturing practice. For this reason, in recent years, academic research has attempted to consider real-life scheduling problems. Standard benchmark problems do not attract the attention of people in industry since practical scheduling problems are far more complex than the famous Fisher and Thompson's [70] MT06, MT10, MT20, and Lawrence's [71] benchmark problems that are still used in most research. Table I gives a summary of results that have been published recently for the three Fisher and Thompson problems. The best and average (wherever available) results of each method are presented. Table II summarizes the results published for some of Lawrence's benchmark problems.

TABLE I
PUBLISHED RESULTS ON FISHER AND THOMPSON'S BENCHMARK PROBLEMS; OPTIMAL VALUES: FT 6×6: 55, FT 10×10: 930, FT 20×5: 1165

	FT 6X6		*FT 10X10*		*FT 20X5*	
PAPERS	***Best***	***Aver.***	***Best***	***Aver.***	***Best.***	***Aver.***
Aizpuru et al.[41]	-	-	930	951	-	-
Cao et al.[26]	-	-	945	953.5	1176	1198.3
Cho et al.[65]	55	-	943	-	-	-
Croce et al.[33]	55	55	946	965.2	1178	1199
Dorndorf et al.[30]	55	-	938	-	1178	-
Gen et al.[29]	55	-	962	-	1175	-
Gohtoh et al.[68]	-	-	930	935.36	1165	1180.34
Kim et al.(1995)[62]	-	-	930	931.57	1165	1165.97
Kim et al.(1996)[63]	-	-	930	930	1165	1165.27
Kobayashi et al.[58]	-	-	930	934.3	1165	1217.4
Ono et al.[59]	-	-	930	931.1	1165	1176.5
Park et al.[61]	-	-	936	949	1178	1185
Shi et al.[46]	-	-	930	946.2	1165	-
Yamada et al.[64]	-	-	930	934.5	1165	1177.3

TABLE II
PUBLISHED RESULTS ON LAWRENCE'S BENCHMARK PROBLEMS; (*) DENOTES OPTIMAL VALUE

	Aizpuru et al. [41]		*Cao et al. [26]*		*Kim et al. [62]*		*Croce et al. [33]*		*Park et al. [61]*	
TEST NO.	Best	Aver.	Best	Aver.	Best	Aver.	Best	Aver.	Best	Aver.
LA01			666*	666	666*	666	666*	666		
LA06			926*	926	926*	926	926*	926		
LA11			1222*	1222	1222*	1222	1222*	1222		
LA16			956	980	945*	945.4	979	989		
LA21	1056		1061	1083.6	1055	1055.8	1097	1113.6		
LA22					935	935.47			935	949
LA26			1227	1231.2	1218*	1218	1231	1248		
LA27	1255				1255	1264.9				
LA31			1784*	1784	1784*	1784	1784*	1784		
LA36			1337	1348			1305	1330.4		

However, a considerable number of recently published papers address real-life scheduling cases. Herrmann *et al.* [48] described the development of a global scheduling system for a semiconductor test area. Niemeyer and Shiroma [54] used EA's for the scheduling of factories of a multinational company. Terano *et al.* [35] combined EA's and SA for a scheduling problem in plastic injection moulding. Gilkinson *et al.* [34] tackled the scheduling problem of a company that produces laminated paper and foil products. Hamada *et al.* [72] approached a complex scheduling problem in a steel-making company using a hybrid system based on EA's and expert systems. Shaw and Fleming [44] and Kumar and Srinivasan [36] proposed evolutionary computation methods for the solution of scheduling problems in companies that produce ready-chill meals and defense products, respectively. Finally, Sakawa *et al.* [73] considered the scheduling problem of a machining center using an evolutionary algorithm.

III. The Flow-Shop Scheduling Problem

A. Introduction

The permutation flow-shop scheduling problem, or the job-sequencing problem as it is often called, is another manufacturing optimization problem that attracts particular research interest. It is relatively easy to apply evolutionary computation methods to this problem since it can be formulated as a classic TSP with path representation. This latter problem has been a subject of research from the early days of evolutionary computation. As a result, the efficient operators that have been developed for the TSP are directly applicable to the flow-shop scheduling problem.

B. Problem Formulation

The permutation flow-shop scheduling problem involves ordering n jobs to be processed in m machines. The difference between the job-shop and the flow-shop scheduling problem is that, in the latter case, each job undergoes the same machining sequence, while the sequence of operations is the same on each machine. This means that the solution of the problem can be represented as a permutation of all jobs to be processed:

$$[J_1, J_2, J_3, \cdots, J_n]$$

where n is the total number of jobs. The conditions that were introduced for the JSSP hold for the flow-shop scheduling problem as well. The minimization of makespan is usually employed as the objective of the scheduling algorithm. The problem is also known as the $n/m/P/C_{\max}$ problem [21]. In the special case of $m = 1$, the problem is described as the one-machine scheduling problem. Garey *et al.* [23] have shown that the problem is NP hard in the strong sense (proof by transformation of the 3-PARTITION problem to the associated flow-shop decision problem).

1) Variations from the Basic Form: The minimization of makespan is used as the main objective in a number of papers reviewed in this section [75]–[79]. However, in recent years, more complicated formulations of the problem have been considered, with various alternative optimization criteria included. Murata *et al.* [80] used a multiobjective GA (MOGA) approach for a flow-shop scheduling problem, aiming to simultaneously minimize makespan, total tardiness, and total flow time of the production. Minimization of total tardiness was also employed as an optimization criterion by Lam *et al.* [81]. Sikora [82] attempted to minimize makespan, holding costs (earliness), and overtime (tardiness) in a flow line with limited buffer capacity. Sannomiya and Iima [83], [84] also tried to minimize makespan, at the same time keeping the processing rate of each product as constant as possible. Their formulation of the problem considered the existence of a carrier that transferred products between the machines. Lee and Choi [85], [86] assigned earliness and tardiness penalty weights to schedules for a one-machine scheduling problem. Lee *et al.* [87] presented an interesting formulation of the problem, introducing the concept of a flexible flow line with variable lot sizes. In this case, jobs consisted of splitable lots, and an efficient EA was used to simultaneously optimize the ordering of jobs and the lot sizing. Gonzalez *et al.* [88] considered the "no-wait" version of the job-sequencing problem, where once the processing of a job has started in the first machine of the production line, there must be no time delay between the consequent operations of the job in the following machines. An EA enhanced with heuristic methods was used for the solution of the problem. Herrman and Lee [89] described a class-one machine scheduling problem, where jobs belonged to different classes, with each class having sequence-dependent setup times. The evolutionary algorithm that they presented generated different input conditions for a minimum waste heuristic algorithm which accomplished the task of producing legal schedules. Karabati and Kouvelis [90] addressed the flow-shop scheduling problem with controllable processing times, i.e., the problem where the processing time of a part is not fixed, but can assume a number of different values. An EA was employed for the solution of large-scale problems of this type. Finally, Ishibuchi *et al.* [91] proposed a fuzzy mathematical formulation of the problem, using the concept of fuzzy due dates. The optimization criteria were the maximization of the minimum satisfaction grade and the maximization of the total satisfaction grade.

C. Encoding

The permutation representation is used in most of the papers surveyed in this section. A permutation is a natural representation for the solution of the problem since there are many well-tested operators to ensure the feasibility of solutions and to enhance the evolutionary process.

There are, however, some exceptions to this rule. The most notable is that of Lam *et al.* [81], who introduced a pigeon-hole coding scheme. In this representation, the value of each gene corresponded to the index of the job selected for scheduling, out of the list of unscheduled jobs. Each time a job was scheduled, the list of unscheduled jobs was reindexed, and the value of the next gene defined the job selected out of the new set. Their representation allowed the use of traditional crossover and mutation operators without producing infeasible solutions. Some slight modifications in the encoding of solutions were also present in [82] and [87], in order to accommodate the simultaneous lot sizing that was attempted by these algorithms. Lee and Choi [85], [86] used parallel genetic algorithms (PGA's) with binary representation for one machine scheduling problems. Finally, Kebbe *et al.* [92] adopted the vibrational-potential method (VPM) for the solution of sequencing problems. VPM is an evolutionary computation method based on the concept of information propagation in nature, which employs different representation schemes.

D. Operators

1) Crossover Operator: There is a long-running debate about the suitability of particular crossover operators for sequencing problems. Michalewicz [11] argued that the flow-shop scheduling problem has certain characteristics that distinguish it from the TSP; thus, the suitability of a particular operator for the TSP is not necessarily valid for all sequencing problems. Since most of the crossover operators have been developed for the TSP, it is easy to understand that the selection of a crossover operator for flow-shop scheduling is not straightforward.

The preservations of order, position, and adjacency of genes are the main characteristics of an operator for sequencing problems. One of the earliest crossover operators is order crossover (OX) [93]. OX preserves the relative order of jobs from the parents. Katoh *et al.* [94] used OX for the solution of a one-machine scheduling problem with uncertain processing times. Davis also introduced the uniform order-based crossover operator [95], which was adopted by Drake and Choudry [78] and by Lee and Choi [85] in their attempts to solve job-sequencing problems. Uniform order-based crossover preserves the absolute position and relative order of jobs from the parents. Researchers have proposed many variations of OX for the flow-shop scheduling problem, like the one- and two-point crossover operators introduced by Murata *et al.* [77], [96]. Sannomiya and Iima [83], [84] and Fichera *et al.* [97] proposed versions of OX. Another well-known TSP operator is the partially mapped crossover operator (PMX) [98], which preserves elements of the absolute order and relative position of jobs from the parent chromosomes. Chen *et al.* [76], [99] used PMX to solve a continuous flow-shop scheduling problem. An interesting element of their approach is that it employed three heuristics (job insertion method (JIB) [100], Campbell, Dudec, and Smith's (CDS) heuristic, and Dannenbring's heuristic) for the initialization of the population. Shridhar and Rajendran [101], [102] also used the PMX operator together with their own DELTA operator (which determines the selection policy of the algorithm) in order to obtain optimal schedules for a flow-line-based manufacturing cell. The same operator was present in the algorithm proposed by Braglia and Gentili [79], who enhanced the evolutionary process by using a neighborhood search algorithm for fine local tuning.

Ross and Tuson [103] directed the search of various stochastic optimizers using an idle time heuristic, i.e., a heuristic that considered the time a job is waiting on a machine before its processing. An EA with a modified PMX operator was employed as a representative of evolutionary computation methods. However, directed search did not seem to have any significant effect on the performance of the algorithm.

An increasing number of researchers adopt the edge-recombination operator [104] and its enhanced version [105]. It was originally deigned for the TSP, and its main characteristic is that it preserves adjacency information from the parents. It has recently been used by Sikora [82], Lee *et al.* [87], and Gonzalez *et al.* [88] in job-sequencing problems.

Asveren and Molitor [106] proposed two new operators for sequencing problems, namely, the neighborhood relationship operator (NRX) and the meta-ordering operator (MOX). NRX is, in fact, a neighborhood search algorithm, as the MSFX operator proposed by Yamada and Reeves [107]. MSFX has also been used for the solution of the JSSP, as we saw in the previous section. Yaguira and Ibaraki [108] designed a hybrid system based on EA's and dynamic programming (DP) for the solution of one-machine scheduling problems. DP is employed in the crossover phase of the algorithm, leading to an efficient and fast optimization method.

The diversity of the operators used in all previous research papers is a result of the uncertainty that exists about the superiority of a particular operator. The use of independent test problems by each researcher makes the comparison very difficult, as we will discuss in the next paragraph. Murata *et al.* [77], [96] compared the performance of seven crossover operators, and their results highlighted the superiority of two-point crossover operator. On the other hand Lee *et al.* [87] compared the edge-recombination, PMX, CX (cycle crossover) [109], and OX operators, and concluded that the edge-recombination operator produces the best quality solutions.

2) Mutation Operator: The mutation operator is often given little importance in research papers; however, its contribution to the best possible exploration of the solutions' search space is important for the evolutionary process. The most well-known mutation operators for sequencing problems are the "swap" operator, which simply exchanges the position of two randomly selected genes in the sequence, and the "shift" operator, which shifts the position of a gene some places to the left or right. Murata and Ishibuchi [96] investigated the performance of five mutation operators on sequencing problems, and the best results were given by the "shift" operator. Their research also proved that the combined effect of the best crossover and mutation operators is not necessarily positive. Their experimentation showed that the best performance is given by the combined effect of two midperformance operators. This conclusion confirms the difficulty of selecting operators for job-sequencing problems.

3) EA-Parameter Specification: The selection of EA parameters (population size, probability of crossover, probability of mutation, etc.) plays a vital role in the performance of the algorithm. Chen *et al.* [99] proposed the introduction of a meta-level EA [110] which optimally controlled the values of parameters, while the evolutionary process was in progress. Pakath and Zaveri [111] also presented an algorithm for the on-line specification of parameters in a GA scheduler.

E. Test Problems and Case Studies

It is extremely difficult to compare the performance of different evolutionary algorithms for flow-shop scheduling since most researchers use their own instances of test problems, i.e., problems where the processing times and due dates of the jobs are selected randomly out of a uniform distribution. Since these instances are not published in detail, they are rarely used by other researchers. A comparison of results taken from this type of problems would not be valid.

Most of the papers referenced in this section use their own problem instances or test problems not widely available. The only exceptions are Reeves [75], Yamada and Reeves [107], and Ross and Tuson [103], who presented results on standard benchmark problems taken from Tailard [112]. Lee *et al.* [87] and Sikora [82] considered the scheduling of a manufacturing plant producing printed circuit boards (PCB's) as a case study.

IV. The Dynamic Scheduling Problem and Comparisons Between Different Scheduling Algorithms

A. Dynamic Scheduling

The cases that we have considered so far in job-shop and flow-shop scheduling addressed static scheduling problems, i.e., problems where the dynamic nature of the scheduling decision is not examined. However, in practical scheduling, a scheduler often has to react to unexpected events. The main uncertainties encountered in a real manufacturing system are the following:

- machine breakdowns including uncertain repair times;
- increased priority of jobs;
- change in due dates;
- order cancellations.

Whenever an unexpected event happens in a manufacturing plant, a scheduling decision must be made in real time about the possible reordering of jobs. This process is known as "rescheduling." The main objective of rescheduling is "to find immediate solutions to problems resulting from disturbances in the production system" [125].

Until recently, evolutionary computation methods have rarely been used for dynamic scheduling, due to their inability to cope with real-time decision making. They were developed and tested on static scheduling problems that did not require real-time control. However, in the last few years, EA's have been employed as parts of hybrid dynamic scheduling systems, which exploit their useful characteristics.

1) Machine Learning Methods: Machine learning is one of the methods that have traditionally been used in manufacturing environments to face uncertainties. Chiu and Yih [113] proposed such a learning-based methodology for dynamic scheduling. They divided the scheduling process in a series of ordered scheduling points. An evolutionary algorithm examined which dispatching rules performed better for each of these points, given a set of plant conditions (system status). The chromosome was formed by a series of genes, each one representing a respective scheduling point and taking as a value one of the available dispatching rules. The performance of the algorithm was simulated under different plant conditions, forming a knowledge base that described the scheduling rules that were preferable in different cases. A binary decision tree was used to describe the gained knowledge. This method had the advantage of being able to modify its existing knowledge (new system conditions), without having to reconstruct the entire knowledge base. Aytug *et al.* [114] presented a different machine learning approach for dynamic scheduling, based on classifier systems [9]. In this case, an initial knowledge base was given, and an EA modified it, using results taken from the simulation of the production line. In that way, the system learned to react to certain unexpected events. A hybrid system based on neural networks, EA's, and an inductive learning algorithm called trace-driven knowledge acquisition (TDKA) [115] was used by Jones *et al.* [116]–[118] to infer knowledge about the scheduling process. A back-propagation neural network selected a number of candidate dispatching rules out of a larger set of available rules. The schedules formed by these dispatching rules were used as the initial population of an EA that evolved an optimal schedule. The results taken from the simulation of the schedule helped TDKA to create a set of rules that formed the knowledge base. Lee *et al.* [119] proposed a hybrid scheduling framework which consisted of an inductive learning system for job releasing in the plant, and an EA-based system for the dispatching of jobs at the machines. The genetics-based machine learning (GBML) method of Goldberg [5] and an EA-based status selection method have also been employed by Tamaki *et al.* [120] and Ikkai *et al.* [121], respectively, to induce scheduling knowledge from manufacturing systems.

2) Alternative Methods: Fang and Xi [122] presented a different rescheduling strategy based on the rolling horizon optimization method. Scheduling was performed periodically on a predefined number of jobs that formed the "job window." Rescheduling was initiated either by the elapse of a job window or by the occurrence of an unexpected event. An EA evolved an optimal schedule for each planning horizon, considering the status of the system. The same concept of job windows was adopted by Cartwright and Tuson [123], who employed an EA to dynamically control the scheduling of a chemical flow-shop. Bierwirth *et al.* [124] proposed a similar approach, aiming to decompose a nondeterministic job-shop problem in a series of deterministic smaller ones. Each subproblem was then solved with the help of the static scheduling EA method that we described in the JSSP section [19].

Finally, Jain and Elmaraghy [125] presented a steady-state EA framework for the scheduling of an FMS system. Specially designed algorithms dealt with unexpected events like machine breakdowns and order cancellations. A series of test cases indicated the validity of the method for scheduling and rescheduling purposes.

B. Comparison of Different Heuristic Methods for Scheduling Optimization

Evolutionary computation is not the only nonanalytical optimization method that has been proposed for the solution of scheduling problems. Iterative improvement techniques, random search techniques, simulated annealing, tabu search, and hybrid techniques are some well-known scheduling optimization methods. Tsang [126] presented an overview of OR (operations research) and AI (artificial intelligence) methods that have been used for scheduling problems. Many researchers have attempted to compare the performance of these optimization methods, and results were recently published in a series of papers.

Dorn *et al.* [127] compared the performance of iterative deepening [128], random search, tabu search, and EA's on the scheduling of a steel manufacturing plant in Austria. Iterative deepening and tabu search produced the best results for this particular case study. The same techniques, with the addition of a hybrid EA–local search method and simulated annealing, were tested on a one-machine scheduling problem by Yaguira and Ibaraki [129]. Their conclusion was that, while local search techniques were computationally efficient and produced moderate solutions, simulated annealing and genetic local search performed much better, but introduced a significant computational overhead. The one-machine scheduling problem was also used for the comparison of local search, simulated annealing, tabu search, and EA's by McMahon and Hadinoto [130]. Simulated annealing gave the best performance, both in numerical and computational results.

In all previous comparisons, specific representations and genetic operators were used by individual researchers; thus, generalization of the conclusions would not be valid. On the other hand, there are indications that the evolutionary process is greatly enhanced when it is hybridized with local search techniques. Additional evidence was given by Glass and Potts [131], who compared the performance of multistart descent, threshold accepting, simulated annealing, tabu search and EA's in a number of flow-shop scheduling problems. While the performance of EA's was poor initially, it was greatly improved when the algorithm was hybridized with a local search method. The same results were reported by Ishibuchi *et al.* [91] for their fuzzy flow-shop scheduling problem discussed in the previous section. They compared the performance of multistart descent, simulated annealing, tabu search, and EA's, and while tabu search outperformed all other individual optimization methods, a hybrid multistart descent–EA system performed equally well.

Up-to-date developments in evolutionary computation approaches to scheduling and time-tabling can be found in EVOSTIM (EVONET working group on scheduling and time-tabling) dynamic report, available through WWW [74].

V. Process Planning

A. Introduction

Process planning is one of the most complex manufacturing phases. It comprises a series of tasks that are heavily dependent on the type of product that is to be processed. Process planning takes as input the design characteristics of a product (CAD files), and gives as output its complete production plan. This plan should determine the machining processes needed, the tools to be used, and the sequencing of operations. If more than one plan is available, then an optimal plan should be selected.

Process planning can be more or less elaborate, according to the processing requirements of a particular part. Horvath *et al.* [132] illustrated some elements of the process that should be determined by a process plan.

Process planning is the link between the design and manufacturing phase of a product. The design phase is highly automated nowadays with the introduction of state-of-the-art computer-aided design (CAD) programs. However, computer-aided process planning (CAPP) programs are not so highly developed, and research interest in the field is growing. An excellent review of CAPP methods can be found in [132].

B. Operation Sequencing

Operation sequencing is an important task of process planning. The planner must determine the machining sequence of parts, taking into account all of the existing precedence constraints for the machining of features. These constraints are normally given in the form of a precedence graph. Usher and Bowden [133] proposed an evolutionary computation approach for the solution of this problem. The number of genes in the solution was equal to the number of features that must be machined. There was a special decoding procedure based on the feature precedence graph, which transformed any string into a feasible sequence of machining operations. This representation was first introduced by Yip-Hoi and Dutta [134]. The total number of setups, the continuity of motion, and the loose precedence determined the quality of solutions. Takatori *et al.* [135] adopted a TSP representation for the solution of the same problem, using a repair mechanism to cope with solutions that violated the constraints. The objectives of their algorithm were the minimization of the total change cost, the machining cost, and the nonmachining cost.

Kamhawi *et al.* [136] developed an elaborate feature-sequencing system based on EA's. The representation scheme was the same as in [135], but the evaluation of solutions was based on rules and constraints about safety, quality, and minimization of tool changes and tool travel. The user assigned a weight to each of these objectives, according to his preferences.

Norman and Bean [137] discussed the problem of operation sequencing and tool allocation in parallel machine tools (PMT). A PMT is a machine capable of processing more than one part at a time since it contains multiple spindles. A random-keys coded EA was proposed for the solution of the problem. The tool allocation task was dealt with in the introduction of an integer part to the value of the genes. This part defined the machining unit (MU) which was responsible for a particular operation. The decimal part of the value determined the sequence of operations. The authors also proposed the enhancement of the algorithm with a heuristic method, presenting results that justified their decision. Yip-Hoi and Dutta [134] tackled the same problem using an efficient solution representation based on feature precedence graphs, as was discussed earlier. The objective of their algorithm was the minimization of the part's total processing time.

C. The Optimal Plan Selection Problem

The optimal plan selection problem is the task of selecting an optimal process plan out of a population of alternative plans. The problem is usually modeled with the help of flow networks, i.e., a construction of arcs and nodes that determines alternative sequences of machining for a given product; see Fig. 1. Each stage of this graph represents a machining operation, and the nodes denote the number of alternative machines that are capable of performing this operation. The weighted arcs define the cost of following a particular machining sequence.

Awadh *et al.* [138] presented one of the first evolutionary algorithms for the solution of the optimal plan selection problem. Each stage of a process plan was represented by a binary-coded matrix, where the occurrence of a bit with positive value denoted the presence of a connection between the corresponding nodes of the matrix. The authors warned that this representation could sometimes lead to the existence of more than one processing plan for a single chromosome solution. A decoding algorithm called "path modifier" ensured that there was a "1-to-1" relationship between the genotype and the phenotype of each solution. The objective of their approach was the minimization of the overall cost. Zhou and Gen [139] noted that fast and efficient algorithms, like the shortest path method and dynamic programming, are capable of producing good solutions for single-objective process planning problems like the previous one. They argued that evolutionary computation methods are ideal for the multiobjective version of the problem, which cannot be easily expressed as a shortest path or dynamic programming problem. The authors constructed an EA that used the same network flow model, but had an efficient integer solution representation that did not require the existence of additional operators like the "path modifier."

D. Advanced Process Planning Methodologies

Concurrent engineering has received much attention lately as a modern approach to manufacturing optimization. It is a manufacturing philosophy where the design and the related manufacturing processes of a product are integrated into one procedure (the reader should refer to Singh [140] for an overview of concurrent engineering). Process planning and scheduling are two manufacturing processes that are closely related. One of the aspects of concurrent engineering is the integrated process planning (in terms of the optimal selection of a process plan) and scheduling of a product. Husbands [141] and McIlhaga *et al.* [142] proposed an EA-based method for the simultaneous determination of planning and scheduling in a vehicle manufacturing company. They used a distributed genetic algorithm (DGA) [143] approach with a diploid chromosome representation that defined both the sequencing of operations and the use of alternative machines. A number of different optimization objectives were included, such as the minimization of makespan, flow time, and tardiness.

Bowden and Bullington [144] created a hybrid system called GUARDS, based on unsupervised machine learning and EA's in order to optimize the control of a manufacturing process. The system learned to select the optimal process plan according to the status of the plant. GUARDS was an extension of the well-known SAMUEL system [145].

Horvath *et al.* [132] described a complete process planning procedure, from the input of part specifications in the form of CAD files, to the optimization of the constructed process plan. They used an object-oriented approach in the form of "features."

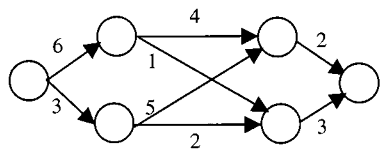

Fig. 1. Representation of a process plan.

A “feature” was an object that defined specific operations and contained all of the relative functional, geometrical, and technological data. Knowledge-based reasoning was used for the generation of plans, which were then optimized with the help of a genetic algorithm. Zhang *et al.* [146] recently developed a similar complete CAPP system for parts manufactured in job-shop environments. They adopted a direct solution representation, as was introduced by Burns [39]. Each chromosome defined the sequencing of operations, machine-tool assignments, and tool approach directions (TAD’s) for an individual process plan. In this way, the procedures of operation sequencing and process plan selection were integrated.

Hayashi *et al.* [147] introduced an interesting method for the evaluation of future plans in a manufacturing plant with uncertain parameter values. A binary-coded EA was employed for the evaluation task. The solution was represented by a string of all of the plant’s parameters, and the objective of the algorithm was defined according to user’s preferences.

VI. Optimization Problems in Cellular Manufacturing

A. Introduction

Cellular manufacturing is the application of group technology (GT) in manufacturing systems. GT was first introduced in the former USSR by Mitrofanov [148], and was popularized in the West by Burbidge [149], who introduced production flow analysis (PFA), the first scientific method for creating manufacturing cells. Cellular manufacturing is a manufacturing philosophy that attempts to convert a manufacturing system into a number of cells. Each cell manufactures products with similar processing characteristics. Ideally, all of the processing operations of a part should be completed within a cell. However, in realistic cases, intercell movements of parts are always present. Cellular manufacturing offers certain advantages to midvariety, midvolume production lines like the reduction of setup and transfer costs, the minimization of inventory, improved quality, and significant savings in plant space.

A vast bibliography exists on the subject of cellular manufacturing. A good introduction is given by Burbidge [150], and a critical review of up-to-date developments can be found in [151]. A considerable number of industries have adopted the concept of cellular manufacturing, as Wemmerlov and Hyer illustrate in a series of papers [152]–[155].

There are three main phases in the design of a manufacturing cell: 1) the grouping of machines into cells, better known as the cell-formation problem, 2) the layout of cells in the plant, and 3) the layout of machines within the cells. The implementation of each of these stages leads to difficult optimization problems, where traditional optimization methods are incapable of finding optimal solutions in reasonable time. In the following paragraphs, we will examine some evolutionary methods that have been used recently to tackle optimization problems associated with cellular manufacturing.

B. Formation of Manufacturing Cells

1) Historical Development: The formation of manufacturing cells is an optimization problem that has been extensively researched over the last 20 years. A considerable number of alternative methods have been proposed for the solution of the problem. Singh [156], Offodile *et al.* [157], and Morad [158] give comprehensive reviews of the problem, and attempt to taxonomize all of these methods into certain categories. An analytical review of the methods is beyond the scope of this paper. However, it is important to reference the most significant of them. As already mentioned, Burbidge [149], the pioneer researcher in cellular manufacturing, introduced the first method of designing manufacturing cells, namely, production flow analysis. His method aimed to create manufacturing cells by a series of manual manipulations on the rows and columns of the machine-component matrix. Some other well-known methods, like rank-order clustering (ROC) [159] and the direct clustering algorithm (DCA) [160], are based on the same matrix. Another well-known cell-formation method is single linkage cluster analysis (SLCA), introduced by McAuley [161], which is based on similarity coefficients between the machines. Coding and classification methods [162], graph partitioning [163], mathematical programming [164], neural networks [165], and fuzzy logic [166] are some other methods that have been proposed for the solution of the cell-formation problem.

2) Evolutionary Computation Methods: Unlike scheduling, the cell-formation problem had not been a subject of evolutionary computation research until very recently. Venugopal and Narendran [167] were the first researchers to approach the cell-formation problem using EA’s. Their objective was the minimization of the intercell traffic and the balancing of load in the cells. A different population of solutions was employed for each of these objectives. The solution representation was simple and efficient. Each machine in the plant corresponded to a gene in the chromosome. The value of the gene defined the owing cell of the respective machine. The total number of cells in the plant was predetermined, but the formulation of the problem considered the processing time of parts, which was a serious improvement in comparison to the traditional cell-formation methods. Gupta *et al.* [168], [169] enhanced this formulation by considering the intracell moves of the parts and the intracell layout. Special care was also taken to ensure that no cell remained empty during the evolutionary process. Billo *et al.* [170] adopted a direct solution representation, based on a two-part chromosome. The first part was a permutation of all parts to be processed, while the second part denoted the cutoff points of the first part. Each segment between cutoff points denoted a part family. The objective of their algorithm was the maximization of machines’ similarity within the cells and the minimization of the total number of cells. The advantage of this method was that the total number of cells was not predefined, but the structure of the chromosomes was quite complex and computationally expensive. However, the algorithm performed well on a series of test problems, including some ill-structured machine-component matrices. Joines *et al.* [171] introduced a new, efficient integer programming formulation of the problem, which reduced the search space significantly. An evolutionary algorithm was employed for the solution of the problem, with the variables of the mathematical formulation coded into the chromosome. Only the upper bound of the total number of cells needed to be specified. The objective of the algorithm was the minimization of exceptional elements and voids (zero's in the diagonal blocks) in the machine-component matrix. The validity of the method was depicted by results on test problems taken from the literature. Su and Hsu [172] used the classic Venugopal solution representation, but their chromosome also accommodated the existence of multiple machines of the same type. Morad and Zalzala [173] proposed the simultaneous optimization of several objectives, using a weighted-sum approach. Pierreval and Plaquin [174] adopted the classic representation scheme, with binary-coded genes. Suer [175] presented a preliminary discussion on the design of part families using evolutionary programming [7]. Dimopoulos and Zalzala [176] proposed an evolutionary algorithm for the cell-formation problem of a pharmaceutical company. Both the representation of the solution and the genetic operators were purpose based. Different multiobjective optimization methods were compared on the solution of the problem.

3) Hybrid Methods: The cell-formation problem is a difficult optimization problem; thus, various methods of enhancing the evolutionary process have been proposed. Paris and Pierreval [177] utilized distributed evolutionary algorithms, attempting to increase the speed of the process in comparison with the methods used so far, which were, in their own words, "notoriously slow." Hwang and Sun [178] formulated the problem using a generalized quadratic assignment mathematical model. The representation of solutions was a permutation of all machines in the plant, each one uniquely identified by a number. A greedy heuristic was employed for the assignment of machines to cells. The authors used a number of comparative measures to evaluate the performance of the method in various test problems. Finally, Zhao *et al.* [179] presented a fuzzy clustering method for the solution of the problem, which took into account the uncertainty and imprecision that usually exist in the problem data. Fuzzy clustering was implemented using an EA that employed fuzzy c partitions as individual chromosomes. This method was a typical example of hybrid systems that exploit the positive characteristics of individual algorithms and result in robust optimization methods.

C. Cell Layout and Machine Layout Optimization Methods

Once the configuration of cells has been determined, the designer must define the layout of machines inside the cells, and the layout of cells in the plant area. These optimization problems belong to the general category of the facility layout problem (FLP). The FLP is a well-known combinatorial problem. It has been formulated as a quadratic set covering the problem, linear integer programming problem, mixed-integer programming problem, and graph-theoretic problem [180]. However, the quadratic assignment problem (QAP) formulation is the most popular in the literature, and since QAP is known to be NP complete (Sahni and Gonzalez [181] by transformation from the Hamiltonian circuit problem) for most problem instances, efficient algorithms must be used for the solution of the problem.

1) Evolutionary Computation Methods for the Solution of the Facility Layout Problem: Several researchers have used evolutionary algorithms to tackle FLP problems in manufacturing. Early approaches can be found in the survey given by Mavridou and Pardalos [182]. Cohoon *et al.* [183] and Tam [184] were the first researchers to approach the problem using evolutionary computation methods. In both cases, the layout was represented by a slicing tree structure (STS, originally introduced by Otten [185]), which can be easily decoded into a layout. A slicing tree is "a binary tree representing the recursive partitioning process of a rectangular area, through cuts. A cut specifies the relative position of departments through four distinguished branching operators" [182]. Kado *et al.* [186] investigated the combination of STS's with different clustering methods for the initialization of the population, and different decoding methods for the creation of layout. Some of these combinations produced improved results on previously published test problems. Garces-Perez *et al.* [187] refined these results by putting the slicing tree structures into a much more natural genetic programming framework, and by employing a variation of one of Kado's most successful decoding method. The STS representation was also adopted by Cheng *et al.* [188] in their GA framework. The authors additionally addressed the issue of the uncertainty of material flow between cells using a convex fuzzy number representation.

Tate and Smith [189] adopted the QAP formulation of the problem, with the objective of minimizing the sum of products of the total material flow and rectilinear distances between the departments. They proposed a flexible-bay layout structure that accommodated unequal sizes for the departments. The plant was divided into a number of bays by end-to-end slices in one direction, and then the bays were split into departments by perpendicular slices. A permutation representation of the solution was used, which determined both the allocation of departments in the layout and the place of bay divisions. Norman and Smith [190] enhanced this representation by using a random-keys EA, thus avoiding feasibility constraints, and by incorporating uncertainty in the mathematical formulation of the problem. Material-handling costs were expressed using expected values and standard deviations for the product volume over time. Suresh *et al.* [191] preferred the permutation representation, but used a much simpler grid structure for the layout. Kazerooni *et al.* [192] proposed an integrated approach for the design of manufacturing cells, which incorporated steps for the simultaneous determination of cell and machine layouts.

Banerjee *et al.* [193] modeled the problem using a mixed-integer programming formulation. They proposed a graph solution representation based on nodes and edges. Nodes corresponded to input–output cell stations, and edges corresponded to material flows between the stations. The layout structure was continuous, and thus much more flexible than the grid and bay structures which restricted the shape of cells. Genetic search was employed as a part of the overall algorithm, aiming to transform the problem into a series of iterative linear programming problems. The robustness of this method was illustrated in a number of test cases taken from the literature, where it was shown to outperform traditional methods.

Conway and Venkataramanan [194] considered an interesting version of the FLP, the dynamic FLP. In this case, the facility layout changes with time, and the algorithm must find the best allocation of facilities over an entire planning horizon. The authors introduced a multipart chromosome representation for the layout, where each part corresponded to a planning period. The position of a gene corresponded to a fixed place in the layout, and the value of the gene denoted the facility that occupied this place for a particular period. The objective of the algorithm was the minimization of layout rearrangements costs and materials flow costs over the entire planning horizon.

2) Special Cases for the Machine Layout Problem: The papers that we have reviewed so far in this section introduced methods that normally apply to the cell layout problem. The machine layout problem is a special type of FLP, and it is usually addressed individually since various assumptions that are made for the FLP are not valid for this problem. Bazargan *et al.* [193] discussed some of these assumptions, such as the equal-sized areas and the *a priori* knowledge of facilities locations. However, elaborate continuous plane FLP methods like [175] can be applied easily to the machine layout problem.

Manufacturing practice usually restricts the search for an optimal intracell layout to a small number of fixed configurations, like the single-row layout, the multirow layout, the semicircled layout, and the loop layout. Braglia and Sternieri [196] utilized an EA in order to find the machine layout in a prefixed single-row structure. The objective of the algorithm was the minimization of the distance traveled by the material-handling device of the cell. The solution was represented by a permutation of all machines in the row. This method performed well in large problem instances in comparison with heuristic approaches. In a similar approach, Braglia and Zavonella [197] adopted the minimization of jobs backtracking as the objective of the algorithm. Braglia [198] also presented an interesting hybrid method, where an EA was employed for the optimization of simulated annealing parameters. Cheng *et al.* [199] addressed the loop machine layout problem using two different objectives: the minimization of the total number of reloads for all products (minsum problem), and the minimization of the maximum number of reloads for all products (minmax problem). The layout was considered to be unidirectional, and there was a single loading–unloading station. The solution was once again represented by a permutation, and the PMX operator was used for crossover purposes. Gen *et al.* [200] introduced a hybrid fuzzy–GA approach for the solution of complex multirow machine layout problems. The objective of the algorithm was the minimization of travel cost between the machines, and the solution was represented by a multipart chromosome that contained information about the total number of rows, the permutation of machines in each row, and the clearances between the machines. Fuzzy sets were used for the representation of the uncertainty that exists in the value of clearances.

Finally, we should note that Bolte and Thoneman [201] addressed the QAP using simulated annealing. The connection of this paper to evolutionary computation is that genetic programming was employed for the optimization of the annealing schedule. The system found good solutions while maintaining acceptable run times. This is one of the few examples where genetic programming has been used for a problem related to manufacturing optimization.

VII. Optimization of Assembly Lines

A. Introduction

Assembly lines are widespread in manufacturing plants. A number of optimization problems are associated with assembly lines, like the assembly sequence planning problem, the sequencing of mixed-model assembly lines, and the assembly line balancing problem. A variety of evolutionary computation methods have recently been proposed for the solution of assembly line optimization problems.

B. The Assembly Sequence Planning Problem

The assembly sequence planning problem (ASSP) has been tackled by Sebaaly and Fujimoto in a number of papers [202]–[205]. It is the problem of finding an optimal sequence of assembling a product that consists of n parts, given its design characteristics. An assembly sequence is feasible if it does not violate the assembly rules and constraints, which are defined by the designer. The authors proposed an evolutionary approach for the solution of this problem, where an individual chromosome is a randomly constructed sequence of parts. An efficient mapping procedure transformed any random assembly sequence into a feasible one. Gropetti and Muscia [206] analyzed the assembly planning procedure, and used an EA in order to obtain a clear contact relational graph.

C. Sequencing in Mixed-Model Assembly Lines

It is often the case that several products with similar characteristics (models) are assembled in a single line (mixed-model assembly lines). The sequencing of models in mixed-model assembly lines is an important task, especially if we wish to apply the JIT principle in the production line. There are a number of objectives associated with this task [207], like the minimization of the line's length, the minimization of total utility work, and the minimization of the variability of parts' consumption (vpc). This latter objective is critical in JIT systems. Leu *et al.* [208] addressed the problem of sequencing a mixed-model assembly line with the objective of minimizing vpc in a JIT production system. An EA was used for the solution of the problem, with each chromosome representing a sequence of models to be assembled. The sequence was cyclic, and the number of individual models in each sequence was fixed. This method performed better on some test problems than the traditional Toyota goal-chasing algorithm (GCA) [209], which is often used in JIT production systems. Kim *et al.* [207] adopted the same representation for the sequencing of a mixed model assembly line, where the objective was the minimization of the total length of the line.

D. The Assembly Line Balancing Problem

Another well-known optimization problem of assembly lines is the assembly line balancing problem. Given n workstations and m parts to be assembled, the assignment of parts to workstations should be defined according to certain optimization criteria. Two versions of the problem are usually considered. The first version aims to minimize the total number of workstations in the plant given a fixed cycle time, while the second version aims to minimize the cycle time, given a fixed number of workstations. Secondary objectives like the minimization of balance delay and the minimization of probability of line stoppage are also considered. Suresh *et al.* [210] presented an excellent literature review on the assembly line balancing problem, and proposed an evolutionary algorithm for the solution of a similar problem, where the objective was mainly the minimization of the smoothness index of balance delay. The solution was represented by a list of sets with length equal to the total number of workstations. Each set contained one or more jobs. All of the initial solutions were feasible, and special operators ensured the feasibility of solutions throughout the evolutionary procedure. The authors also presented an alternative version of the algorithm, where a number of infeasible solutions were allowed in the population. This particular version worked well on large problem instances. Rubinovitz and Levitin's [211] representation was a permutation of all parts, divided into a number of sections equal to the total number of workstations. Initially, random sequences were constructed, and then special mechanisms were employed to reorder the sequences according to the precedence constraints and to divide them into an appropriate number of sections. Tsujimura *et al.* [212] presented an interesting EA–fuzzy logic method for solving the assembly line balancing problem, aiming to minimize the balance delay. The solution representation was a classic permutation that considered all precedence constraints. The processing time of each job was not deterministic, but was defined by a fuzzy set. The allocation of jobs to workstations was accomplished using the EA sequence, the fuzzy sets, and a standard predefined maximum completion time. Starting with the first job of the sequence, the fuzzy sets of processing times were added, until the upper limit of the sum of fuzzy sets became larger than the predefined maximum completion time. The set of jobs that comprised the sum was assigned to the first workstation, and the procedure started again from the next job after this set in the sequence. Special mechanisms and operators ensured the feasibility of solutions.

E. Secondary Assembly Line Optimization Problems

A number of secondary optimization problems in assembly lines have also been the subject of evolutionary computation research. Among them are the optimization of buffer sizes between workstations in an assembly system [213], the scheduling of multilevel assemblies [214], and the scheduling of flexible assembly systems [215]. Finally, Watanabe *et al.* [216] have used an EA in order to solve the generalized line balancing problem.

VIII. Design Optimization Problems

A. Introduction

Design is a complicated and time-consuming phase in the development of a product. Although design is most often not directly addressed as a manufacturing optimization problem, it constitutes one of its most critical aspects since it irretrievably constrains the manufacturing process. Every design must be faultless and properly optimized; otherwise, the result will be huge redesign costs. Enormous effort has been devoted to the development of efficient CAD systems in order to simplify and speed up the design process. Evolutionary computation methods have been applied successfully to complex design optimization problems. In the following paragraphs, we will review some of the recent papers in this field.

Traditionally, the design process starts with the creation of a mathematical model for the product that is to be manufactured. The model is then implemented as a computer program, allowing the designer to explore the effects of altering the values of the parameters. This optimization process is usually implemented on a "trial-and-error" basis. The incorporation of EA's in the heart of the design process enhances and automates the procedure of parameter optimization [217].

B. Parameter Optimization Problemss

Cao and Wu [218] adopted an evolutionary programming [7] approach for the solution of a mechanical design optimization problem. A number of design variables needed to be optimized, subject to certain constraints. Continuous, binary, integer, and discrete variables were included in the mathematical model, a condition that made the optimization procedure even harder. The solution was represented by a string of design variables initialized within the constraints, while a special mutation procedure was used for each type of variable. Two design problems were used to illustrate the method: the design of a gear train, and the design of a pressure vessel. The algorithm performed equally well or better in comparison with other optimization methods like the branch and bound algorithm and simulated annealing. Rasheed *et al.* [219] proposed an EA for the solution of a similar parameter optimization problem which involved only continuous variables. The solution was a string of all parameters that needed to be optimized, initialized within their constraints. Feasibility problems were accommodated using a penalty function. The evolutionary process was enhanced with the introduction of two crossover operators, namely, line crossover and guided crossover, which produced an offspring on the line connecting the parent chromosomes, considering the solutions' search space. The algorithm was tested on two complex design optimization problems: the design of a supersonic transport aircraft, and the design of a supersonic missile inlet. The method performed much better on these problems than a classic binary-coded GA and a sequential quadratic programming method. Coello and Christiansen [220] gave a nice extension to the use of EA's for parameter optimization by incorporating the weighted-sum multiobjective optimization method in the heart of the evolutionary process.

It was a realistic extension since conflicting objectives always exist during the design phase of a product. The method was tested on the design of an I beam and a machine tool spindle, considering multiple conflicting objectives.

C. *Advanced Design Optimization Problems*

It is often the case that design optimization problems are quite complicated, involving a series of highly related tasks. In these circumstances, the problem is normally divided into a series of subproblems, each comprised of a certain number of tasks. The optimal decomposition of multidisciplinary optimization problems and the optimal ordering of tasks within each subproblem were considered by Altus *et al.* [221] using an evolutionary algorithm. The objective was the minimization of the total length of feedback lines between the tasks. The representation of the solution was a permutation of all tasks involved, and a break character was used to divide the string, and thus the problem, into a series of subproblems. The system was called AGENDA (a genetic algorithm for decomposition analysis), and it performed well on some decomposition problems taken from the literature.

Thornton and Johnson [222] developed an integrated software tool called CADET (computer-aided design embodiment tool) that supported the embodiment phase of the design process, i.e., the creation of a geometrical model of the product, according to the designer's specifications. CADET took these specifications as input, and found a geometrical model that satisfied the constraints. EA's were proposed as an option in the constraint satisfaction part of the system.

Carlson [223] used a GA to optimally select components for catalog design processes. In catalog design, a system is constructed from off-the-shelf components. The EA solution was a string comprised of all types of components used in the design. The value of a gene determined the component selected out of all possible components available for this type. A penalty function handled the violation of constraints. The applicability of the algorithm was demonstrated using the design of a hydraulic system and the design of a thermal fluid system as case studies.

Finally, Iannuzzi and Sandgren [224] addressed the problem of optimally allocating tolerances on product dimensions in order to minimize the total production costs. The authors tackled the problem using an EA-based method, which showed satisfactory results on a series of test problems.

IX. Manufacturing-Related Optimization Problems

A. *Introduction*

In the previous sections, we reviewed recent papers in the field of evolutionary computation for some standard manufacturing optimization problems. However, these are not the only optimization problems associated with manufacturing. The purpose of this section is to illustrate some recent evolutionary computational developments in various manufacturing areas.

B. *PID and Fuzzy Controllers*

The efficient autotuning of PID controllers is a significant optimization problem in the field of process manufacturing. Despite the fact that the problem has been well researched by control engineers, the traditional Ziegler and Nichols tuning rules [225] are still being used in practice. Evolutionary computation methods provide the means of efficient tuning since a solution representation based on PID parameters can be constructed easily. This potential has been recognized by a considerable number of researchers who have used evolutionary algorithms to tune PID controllers. Jones and Oliveira [226] built an EA-based system that initially identified the process model, and then used this model to tune the parameters of the controller off line. The same authors [227] proposed an evolutionary technique for the design of robust SISO Smith predictor PID controllers. Jones and Porter [228] tuned the parameters of a digital PID controller using an evolutionary algorithm. A coevolutionary model was proposed by Jones *et al.* [229] for the design of robust PID controllers. Krohling [230] presented an EA which optimized a PID controller for disturbance rejection. Vlachos *et al.* [231] extended the concept of genetic tuning to PI controllers for multivariable processes. The parameters of all controllers were simultaneously coded in one solution representation. Salami and Cain [232], [233] introduced a hardware-implemented GA (GA processor) which tuned the parameter of a PID controller. The authors also presented encouraging results taken from experiments with multiple GA processors.

Qi and Chin [234] used an evolutionary algorithm to optimally tune the parameters of a fuzzy logic controller (FLC) which had been designed for high-order processes. Kim and Ziegler [235] addressed the same problem using hierarchical distributed GA's (HDGA's). HDGA's are multilevel hierarchical systems composed of local hybrid EA's–expert system structures. These structures are organized in levels, and in each level, the problem is solved on a different degree of abstraction. The advantage of the system was its ability to dynamically change its structure in order to explore promising regions of the search space. Tarng *et al.* [236] employed a binary-coded GA for the design of an optimal fuzzy logic controller which was used in tuning operations.

C. *Process Model Identification*

The identification of process models is essential for the optimal control of manufacturing systems. Polheim and Marenback [237] used genetic programming in order to identify the model of a manufacturing process. Common control engineering tools, like transfer function blocks, were used for the creation of trees (programs). In this way, the algorithm provided structured process models, giving the control engineer a useful insight into a system's internal configuration. Test problems validated the performance of the method, and especially its ability to generalize. McCay *et al.* [238] also employed genetic programming for system identification, constructing the trees with common mathematical functions. There are also a number of researchers who addressed the problem of system identification using EA's. Among them, Reeves *et al.* [239] proposed an interesting method where the solution was coded in terms of the radii and angles of poles and zeros of the transfer function. The values of these variables were constrained within the stability regions; thus, the final solution was guaranteed to be stable.

D. Machine Failure and Maintenance

The failure of machines in the plant is inevitable. Shop-floor engineers aim to diagnose the failure of a machine as quickly as possible. They normally use a number of symptom parameters that are sensitive to changes of specific signals from the plant. Chen *et al.* [240] described some of these parameters, and proposed an evolutionary approach for the determination of an optimal sequence of symptom parameters. Their method resembled genetic programming, in terms of the tree structures that were used as individual chromosomes. Petrovic and Ivanovic [241] presented a hybrid method for machine-noise diagnosis, based on neural networks, expert systems, and EA's. Guzman and Kramer [242] developed a hybrid Bayesian network–EA system that performed on-line monitoring and failure diagnosis based on data taken from the plant.

Maintenance scheduling is another important operation in the shop floor since the disruption of the production process must be as small as possible, but at the same time, the machines must work without failures for the longest time possible. Kim *et al.* [243] proposed an interesting hybrid of EA's and simulated annealing for optimal maintenance scheduling. The acceptance probability of simulated annealing was used for the survival of the less fit offspring in the population.

E. Quality Control

Quality control is an important aspect of modern manufacturing. The optimal allocation of inspection stations in the plant ensures that products are manufactured according to the quality criteria set by the management team. Viswanadham *et al.* [244] addressed this problem in a multistage manufacturing system, and employed an evolutionary algorithm to optimally locate inspection stations. The solution was binary coded, with each gene representing a manufacturing stage. The presence of a station at a particular stage was denoted by a positive value. Patro and Kolarik [245] designed a system that performed statistical processing control using neural networks and evolutionary computation. The neural network identified the process model, and the evolutionary algorithm adjusted the control parameters in order to obtain the desired quality performance. Lu *et al.* [246] presented an EA-based system that optimized the motion of a coordinate-measuring machine used in inspection systems. A permutation representation was employed for the solution of the problem, with each gene corresponding to a testing point that the measuring machine must visit. The algorithm aimed to find the optimal sequence of visiting points that minimized the total length of the inspection path.

F. Advanced Manufacturing Optimization Problems

This section discusses some advanced manufacturing optimization problems that have been the subject of evolutionary computation research. Mak and Wong [247] considered the problem of designing an optimal integrated production–inventory–distribution system, aiming to minimize the overall costs, including inventory holding costs, delivery costs, manufacturing costs, and shortage costs. An evolutionary algorithm was employed for the solution of the problem. Integer programming formulation was adopted, and the solution was represented using the variables of the model. Disney *et al.* [248] addressed the problem of controlling a production and inventory system. Transfer functions were used for the modeling of the problem, illustrated in the form of block diagrams. The solution of the problem was represented by the variables of the transfer function, and a fitness measure was designed based on stock reduction, production robustness, and inventory recovery.

A difficult decision that the marketing team often has to face is the location of inventory centers for the accommodation of department stores, and the allocation of an inventory center to each of these stores. This location–allocation problem was formulated as a nonlinear mixed-integer programming problem, and was solved by Gong *et al.* [249] using an evolutionary approach for the location task and a Lagrangian relaxation method for the allocation task.

Aggregate production planning is a high-level decision-making procedure that takes product capacities and forecast demands as input, and produces aggregate production plans. Stockton and Quinn [250] addressed this problem using a binary-coded GA. The algorithm determined the amount of resources needed each month in order to meet the demand. The resources were expressed in the form of overtime, subcontracts, and stock. Wang and Fang [251] formulated the same problem using a fuzzy linear programming model. They employed Zimmerman's tolerance approach to transform the problem into a linear programming model. The variables of the model formed the chromosome of an evolutionary algorithm that was used for the solution of the problem. Feng *et al.* [252] addressed the problem of joint marketing/production decision making aiming to maximize the net profit of a company. The decision problem consisted of the promotion problem for the marketing department and the production problem for the manufacturing department. Each problem was formulated mathematically, and a respective number of EA's were employed to find optimal solutions. The decision variables of the mathematical models were used for the representation of solutions. Garavelli *et al.* [253] considered the production planning problem of a multinational company that owns manufacturing plants all over the world. Parameters like local market demands and independent capacities must be taken into account in the formulation of the problem. An EA defined which plants would be activated for production and the timing of activation.

The dynamic lot-sizing problem in a multistage, multi-item production system was described by Jinxing [254]. He proposed an evolutionary programming approach with binary representation for the solution of the problem. The objective was the minimization of setup, production, and inventory costs.

G. Various Applications

Xia and Macchietto [255] tackled the problem of optimal design and synthesis of chemical batch plants using a stochastic optimizer called EASY, which was a combination of EA's and simulated annealing. The batch-scheduling problem was also addressed by Morad and Zalzala [173] with the help of an evolutionary algorithm.

All of the equipment in the plant is interconnected to various kinds of pipes. The plant pipe-route optimization problem aims to find the minimum length of pipe interconnections in the plant that satisfies all of the requirements. Kim *et al.* [256] employed a GA with Steiner points representation for the solution of the problem. Their method worked well on a series of test problems.

In a pull (JIT) production system, the demand must always be satisfied without the help of excessive stocks. The total number of kanbans in the plant and the corresponding production trigger values should be optimally defined in order to achieve this objective. Bowden *et al.* [257] addressed this problem using an evolutionary algorithm seeded with the optimal solution of the Toyota equation [191]. Rao and Gu [258] developed a new entropic measure which determined the optimal timing for reconfiguration in a manufacturing plant, and presented a genetic framework for the design of manufacturing systems. Kubota *et al.* [259] described an application of their virus evolutionary GA (VEGA) approach to the self-organization of a cellular manufacturing system. The same problem was tackled by Kawauchi *et al.* [260] with the help of a conventional evolutionary algorithm. Zhao *et al.* [261] addressed the problem of robot selection and workstation assignment in a computer-integrated manufacturing (CIM) system. A bin-packing formulation of the problem was proposed, and an EA was employed for the solution of the problem. A diploid chromosome that accommodated both parts of the problem represented the solution. Recently, McIlhaga [262] designed a framework for solving generic scheduling problems, i.e., scheduling problems of a nonspecific form. This framework was based on DGA's, and was able to solve problems of this kind more efficiently than random search and dispatching rules. The parameters of the problem were defined by the user through a scheduling description language (SDL).

Finally, evolutionary computation applications have been reported for the problems of vehicle distribution scheduling [263], warehouse scheduling [264], sequencing optimization in automotive manufacturing industries [265], optimal control of spinning processes [266], design of flexible electronic assembly systems [267], optimization of textile processes [268], and optimization of area loss in flat glass cutting [269].

X. Conclusion

The use of evolutionary computation methods for manufacturing optimization is expanding. The number of papers published is increasing rapidly, and research covers a wide range of manufacturing problems. The amount of work itself indicates that evolutionary computation methods have established themselves as a useful optimization technique in the manufacturing field, despite the fact that their theoretical foundations are still debated.

Evolutionary computation research has been criticized for the consideration of artificial test problems that are much simpler than real-life manufacturing cases. Our study shows that researchers have reacted to this criticism by considering realistic problems taken from manufacturing plants. This move has also been triggered by the low response of evolutionary computation in manufacturing practice. It is encouraging to report recent projects where companies have adopted evolutionary computation methods in their plants. The gap between academic research and manufacturing practice is not a problem restricted to the field of evolutionary computation. However, in our case, there are a number of additional reasons that make the approach more difficult, and published results indicate that this is usually the case.

- The terminology of evolutionary computation is vague for the manufacturing engineer. Despite the fact that the driving logic of evolutionary algorithms is amazingly simple and efficient, the terminology inherited from genetics predisposes manufacturing engineers to think the opposite.
- Evolutionary computation is a relatively new technique, evolving to deal with more complex real-life problems. There are no universally accepted methods for the determination of technical parameters like population size, probability of applying operators, etc. There is also no guarantee that an algorithm will converge to an optimal or near-optimal solution, except under specific problem conditions.
- There is no standard evolutionary computation toolkit that can be used easily by manufacturing people who are not familiar with evolutionary concepts.

On the other hand, evolutionary computation methods offer solutions that combine computational efficiency and good performance. This significant feature will certainly continue to attract the interest of engineers.

The robustness of evolutionary algorithms is greatly enhanced when they are hybridized with other optimization methods like local search techniques, simulated annealing, tabu search, neural networks, and fuzzy systems. The number of papers introducing hybrid systems is growing, indicating that there is a trend toward this direction.

Evolutionary computation techniques are useful in a series of manufacturing problems, and it is the authors' hope that more research in this area will lead to an increased implementation of real-life manufacturing optimization systems.

Acknowledgment

The authors would like to thank the reviewers for their helpful comments.

References

[1] G. E. P. Box, "Evolutionary operation: A method for increasing industrial productivity," *J. Roy. Statist. Soc., C,* vol. 6, no. 2, pp. 81–101, 1957.
[2] R. M. Friedberg, "A learning machine: Part I," *IBM J. Res. Develop.*, vol. 2, no. 1, pp. 2–13, 1958.
[3] H. J. Bremermann, "Optimization through evolution and recombination," in *Self-Organizing Systems*, Yacobs, Jacobi, and Goldstein, Eds. Washington, DC: Spartan, 1962, pp. 93–106.
[4] J. H. Holland, *Adaptation in Natural and Artificial Systems*. Ann Arbor, MI: Univ. of Michigan Press, 1975.
[5] D. E. Goldberg, *Genetic Algorithms in Search, Optimization and Machine Learning*. Reading, MA: Addison-Wesley, 1989.
[6] I. Rechenberg, *Evolutionsstrategie: Optimierung technischer Systeme nach Prinzipien der biologischen Evolution*. Stuttgart, Germany: Frommann-Holzboog, 1973.
[7] L. J. Fogel, A. J. Owens, and M. J. Walsch, *Artificial Intelligence Through Simulated Evolution*. New York: Wiley, 1996.

[8] L. B. Booker, D. E. Goldberg, and J. H. Holland, "Classifier systems and genetic algorithms," in *Machine Learning: Paradigms and Methods*, J. G. Carbonnel, Ed. Cambridge, MA: MIT Press/Elsevier, 1989, pp. 235–282.

[9] J. R. Koza, *Genetic Programming: On the Programming of Computers by Means of Natural Selection*. Cambridge, MA: MIT Press, 1992.

[10] T. Bäck, U. Hammel, and H.-P. Schwefel, "Evolutionary computation: Comments on the history and current state," *IEEE Trans. Evol. Comput.*, vol. 1, no. 1, pp. 3–17, 1997.

[11] Z. Michalewicz, *Genetic Algorithms + Data Structures = Evolution Programs*. New York: Springer-Verlag, 1992.

[12] M. Mitchell, *An Introduction to Genetic Algorithms*, Cambridge, MA: MIT Press, 1996.

[13] M. Garey and D. Johnson, *Computers and Intractability: A Guide to the Theory of NP-Completeness*. San Francisco, CA: Freeman, 1979.

[14] S. Kirkpatrick, C. D. Gelatt, Jr., and M. P. Vecchi, "Optimisation by simulated annealing," *Science*, vol. 220, pp. 671–679, 1985.

[15] F. Glover, "Tabu search—Part I," *ORSA J. Comput.*, vol. 1, no. 3, pp. 190–206, 1989.

[16] ——, "Tabu search—Part II," *ORSA J. Comput.*, vol. 2, no. 1, pp. 4–32, 1990.

[17] ——, "Tabu search: A tutorial," *Interfaces*, vol. 20, no. 3, 1990.

[18] L. Davis, "Job shop scheduling with genetic algorithms," in *Proc. 1st Int. Conf. Genetic Algorithms Appl.*, J. J. Grefenstette, Ed. Hillsdale, NJ: Lawrence Erlbaum, 1985, pp. 136–140.

[19] A. K. Jain and H. A. Elmaraghy, "Single process plan scheduling with genetic algorithms," *Prod. Planning Contr.*, vol. 8, no. 4, pp. 363–376, 1997.

[20] T. Yamada and R. Nakano, "A genetic algorithm applicable to large scale job shop problems," in *Proc. 2nd Int. Conf. PPS from Nature*, Männer and Manderick, Eds. Amsterdam, The Netherlands: Elsevier Science, 1992, pp. 281–290.

[21] S. French, *Sequencing and Scheduling: An Introduction to the Mathematics of Job-Shop*. New York: Wiley, 1982.

[22] C. Bierwirth, D. C. Mattfeld, and H. Kopfer, "On permutation representations for scheduling problems," in *Proc. 4th Int. Conf. PPS from Nature*, Voigt, Ebeling, Rechenberg, and Schwefel, Eds. Berlin, Germany: Springer-Verlag, 1996, pp. 310–318.

[23] M. Garey, D. S. Johnson, and R. Sethi, "The complexity of flowshop and jobshop scheduling," *Math. Oper. Res.*, vol. 1, pp. 117–129, 1976.

[24] P. C. Chu and J. E. Beasley, "A genetic algorithm for the generalized assignment problem," *Comput. Oper. Res.*, vol. 24, no. 1, pp. 17–23, 1997.

[25] H. Fang, P. Ross, and D. Corne, "A promising hybrid GA/heuristic approach for open-shop scheduling problems," in *ECAI'94: Proc. 11th European Conf. Art. Intell.*, A. G. Cohn, Ed. Chichester, U.K., 1994, pp. 590–594.

[26] H. Cao, H. Xi, Y. Luo, and S. Yang, "GA with hierarchical evaluation: A framework for solving complex machine scheduling problems in manufacturing," in *GALESIA'97: Conf. Genetic Algorithms in Eng. Syst.: Innovations and Appl.*. Stevenage, U.K.: IEE, 1997, pp. 326–331. IEE Conf. Publ. 446.

[27] S. S. Panwalker, R. A. Dudec, and M. L. Smith, "Sequencing research and the industrial scheduling problem," in *Symp. Theory of Scheduling and Appl.*, S. E. Elmaghraby, Ed. Berlin, Germany: Springer-Verlag, 1973, pp. 29–38.

[28] V. Parunak and W. Fulkerson, "GA's and production scheduling," *Genetic Algorithms Dig.*, vol. 8, no. 8, 1994. (electronic version), Available: ftp://ftp.aic.nrl.navy.mil/pub/galist/digests/v8n8.

[29] M. Gen, Y. Tsujimura, and E. Kubota, "Solving job shop scheduling problems by genetic algorithms," in *Proc. 1994 IEEE Int. Conf. Syst., Man, Cybern*. Piscataway, NJ: IEEE, 1994, pp. 1577–1582.

[30] U. Dorndorf and E. Pesch, "Evolution-based learning in a job-shop scheduling environment," *Comput. Oper. Res.*, vol. 22, no. 1, pp. 177–181, 1995.

[31] K. Mesghouni, S. Hamadi, and P. Borne, "Production job-shop scheduling using genetic algorithms," in *Proc. 1996 IEEE Int. Conf. Syst., Man, Cybern.: Part 2*. Piscataway, NJ: IEEE, 1996, pp. 1519–1524.

[32] H. L. Fang, D. Corne, and P. Ross, "A genetic algorithm for job-shop problems with various schedule quality criteria," in *Evol. Comput., AISB Workshop, Selected Papers*, T. C. Fogarty, Ed. Berlin, Germany: Springer-Verlag, 1996, pp. 39–49. (Lecture Notes in Comput. Sci., 1143).

[33] F. D. Croce, R. Tadei, and G. Volta, "A genetic algorithm for the job-shop problem," *Comput. Oper. Res.*, vol. 22, no. 1, pp. 15–24, 1995.

[34] J. C. Gilkinson, L. C. Rabelo, and B. O. Bush, "A real-world scheduling problem using genetic algorithms," *Comput. Indust. Eng.*, vol. 29, no. 1–4, pp. 177–181, 1995.

[35] K. Terano, Y. Yao, K. Okamoto, Y. Hashimoto, I. Nishikawa, T. Watanabe, and H. Tokumuru, "Application of simulated annealing method and genetic algorithm to scheduling problems in plastic injection molding," in *Proc. Japan/USA Symp. Flexible Automat.*: ASME, 1996, pp. 1225–1228.

[36] N. S. H. Kumar and G. Srinivasan, "A genetic algorithm for job-shop scheduling—A case study," *Comput. Indust.*, vol. 31, no. 2, pp. 155–160, 1996.

[37] C. H. Dagli and S. Sittisathancai, "Genetic neuro-scheduler: A new approach for job shop scheduling," *Int. J. Prod. Econ.*, vol. 41, no. 1–3, pp. 135–145, 1995.

[38] M. Pinedo, *Scheduling: Theory, Algorithms and Systems*. Englewood Cliffs, NJ: Prentice-Hall, 1995.

[39] R. Burns, "Direct chromosome representation and advanced genetic operators for production scheduling," in *Proc. 5th Int. Conf. Genetic Algorithms*, S. Forest, Ed. San Mateo, CA: Morgan Kaufmann, 1993, pp. 352–359.

[40] S. Sittisathancai, C. H. Dagli, and H. C. Lee, "A genetic scheduler for job-shops," in *Proc. ANNIE'94: Art. Neural Networks in Eng.*: ASME Press, 1994, pp. 351–356.

[41] J. R. Z. Aizpuru and J. A. Usunariz, "GA/TS: A hybrid approach for job-shop scheduling in a production system," in *Proc. 7th Portuguese Conf. Art. Intell.—EPIA'95*. Berlin, Germany: Springer-Verlag, 1995, pp. 153–164.

[42] B. Giffler and G. L. Thompson, "Algorithms for solving production scheduling problems," *Oper. Res.*, vol. 8, pp. 487–503, 1969.

[43] R. Cheng, M. Gen, and Y. Tsujimura, "A tutorial survey of job-shop scheduling problems," *Comput. Indust. Eng.*, vol. 30, no. 4, pp. 983–997, 1996.

[44] K. J. Shaw and P. J. Flemming, "Including real-life preferences in genetic algorithms to improve optimization of production schedules," in *GALESIA'97: Proc. Conf. Genetic Algorithms Eng. Syst.: Innovations and Appl.*. Stevenage, U.K.: IEE, 1997, pp. 239–244. IEE Conf. Publ. 446.

[45] ——, "Use of rules and preferences for schedule builders in genetic algorithms for production scheduling," in *Evol. Comput., AISB Workshop*, D. Corne and L. Shapiro, Eds. Berlin, Germany: Springer-Verlag, 1997, pp. 237–250. (Lecture Notes in Comput. Sci., 1305).

[46] G. Shi, "A genetic algorithm applied to a classic job-shop scheduling problem," *Int. J. Syst. Sci.*, vol. 28, no. 1, pp. 25–32, 1997.

[47] ——, "A new encoding scheme for soving job-shop problems by genetic algorithm," in *Proc. 35th Conf. Decision Contr.*: IEEE, 1996, pp. 4395–4400.

[48] J. W. Herrmann, C. Y. Lee, and J. Hinchman, "Global job-shop scheduling with a genetic algorithm," *Prod. Oper. Manage.*, vol. 4, no. 1, pp. 30–45, 1995.

[49] R. Haupt, "A survey of priority rule-based scheduling," *OR Spektrum*, vol. 11, no. 1, pp. 3–16, 1989.

[50] J. H. Blackstone, D. T. Philips, and C. L. Hogg, "A state of the art survey of dispatching rules for manufacturing job shop operations," *Int. J. Prod. Res.*, vol. 20, pp. 27–45, 1982.

[51] K. Caskey and R. L. Storch, "Heterogeneous dispatching rules in job and flow shops," *Prod. Planning Contr.*, vol. 7, no. 4, pp. 351–361, 1996.

[52] H. Fujimoto, C. Lian-Yi, Y. Tanigawa, and K. Iwahashi, "Application of genetic algorithm and simulation to dispatching rule-based FMS," in *Proc. 1995 IEEE Int. Conf. Robotics Automat*. Piscataway, NJ: IEEE, 1995, pp. 190–195.

[53] ——, "FMS scheduling by hybrid approaches using genetic algorithms and simulation," in *GALESIA'95: Proc. Conf. Genetic Algorithms in Eng. Syst.: Innovations and Appl*. London, U.K.: IEE, 1995, pp. 442–447. IEE Conf. Publ. 414.

[54] G. Niemeyer and P. Shiroma, "Production scheduling with genetic algorithms and simulation," in *Proc. 4th Int. Conf. PPS from Nature*, X. Voigt, X. Ebeling, X. Rechenberg, and X. Schwefel, Eds. Berlin, Germany: Springer-Verlag, 1996, pp. 930–939.

[55] J.-U. Kim and Y.-D. Kim, "Simulated annealing and genetic algorithms for scheduling problems with multi-level product structure," *Comput. Oper. Res.*, vol. 23, no. 9, pp. 857–868, 1996.

[56] J. C. Bean, "Genetic algorithms and random-keys for sequencing and optimization," *ORSA J. Comput.*, vol. 6, no. 2, pp. 154–160, 1994.

[57] K. Baker, *Introduction to Sequencing and Scheduling*. New York: Wiley, 1974.

[58] S. Kobayashi, I. Ono, and M. Yamamura, "An efficient genetic algorithm for job-shop scheduling problems," in *Proc. 6th Int. Conf. Genetic Algorithms Appl.*, L. Eshelman, Ed. San Francisco, CA: Morgan Kaufmann, 1995, pp. 506–511.

[59] I. Ono, M. Yamamura, and S. Kobayashi, "A genetic algorithm for job-shop scheduling problems using job-based order crossover," in *Proc. 1996 IEEE Int. Conf. Evol. Comput.*. Piscataway, NJ: IEEE, 1996, pp. 2798–2803.

[60] L.-J. Park and C. H. Park, "Genetic algorithms for job-shop scheduling problems based on two representational schemes," *Electron. Lett.*, vol. 31, no. 3, pp. 205–207, 1995.

[61] ——, "Application of genetic algorithms to job-shop scheduling problems with active-schedule constructive crossover," in *Proc. 1995 IEEE Int. Conf. Syst., Man, Cybern*. Piscataway, NJ: IEEE, 1995, pp. 530–535.

[62] G. H. Kim and C. S. G. Lee, "Genetic reinforcement learning approach to the machine scheduling problem," in *Proc. 1995 IEEE Int. Conf. Robot. Automat*. Piscataway, NJ: IEEE, 1995, pp. 196–201.

[63] ——, "Genetic reinforcement learning for scheduling heterogeneous machines," in *Proc. 1996 IEEE Int. Conf. Robot. Automat*. Piscataway, NJ: IEEE, 1996, pp. 2798–2803.

[64] T. Yamada and R. Nakano, "A genetic algorithm with multi-step crossover for job-shop scheduling problems," in *GALESIA'95: Proc. Conf. Genetic Algorithms in Eng. Syst.: Innovations and Appl*. London, U.K.: IEE, 1995, pp. 146–151. IEE Conf. Publ. 414.

[65] B. J Cho, S. C. Hong, and S. Okoma, "Job-shop scheduling using genetic algorithms," in *Critical Technol.: Proc. 3rd World Congr. Expert Syst*. New York: Cognizant Com. Corp., 1996, pp. 351–358.

[66] S. J. T. Liang and M. Mannion, "Scheduling of a flexible assembly system using genetic algorithms," in *Proc. 11th Int. Conf. Comput.-Aided Prod. Eng*. Bury St. Edmonds, UK: Mech. Eng. Publ., 1995, pp. 173–178.

[67] J. Adams, E. Balos, and D. Zawack, "The shifting bottleneck procedure for job-shop scheduling," *Manage. Sci.*, vol. 34, pp. 391–401, 1988.

[68] T. Gohtoh, K. Ohkura, and K. Ueda, "An application of genetic algorithm with neutral mutations to job-shop scheduling problems," in *Proc. Int. Conf. Advances in Prod. Syst., APMS'96*, Okino, Tamura, and Fujii, Eds. Kyoto, Japan: Kyoto Univ., 1996, pp. 563–568.

[69] K. Ohkura and K. Ueda, "Genetic algorithm with neutral mutation for massively multimodal fuction optimization," in *Proc. 1995 IEEE Int. Conf. Evol. Comput*. Piscataway, NJ: IEEE, 1995, pp. 361–364.

[70] H. Fischer and G. L. Thompson, "Probabilistic learning combinations of local job-shop scheduling rules," in *Industrial Scheduling*, J. F. Muth and G. L. Thompson, Eds. Englewood Cliffs, NJ: Prentice-Hall, 1963, pp. 225–251.

[71] S. Lawrence, *Resource Constrained Project Scheduling: An Experimental Investigation of Heuristic Scheduling Techniques*. Pittsburgh, PA: GSIA, Carnegie Mellon Univ., 1984.

[72] K. Hamada, T. Baba, K. Sato, and M. Yufu, "Hybridising a genetic algorithm with rule-based reasoning for production planning," *IEEE Expert*, vol. 10, no. 5, pp. 60–67, 1995.

[73] M. Sakawa, K. Kosuke, and T. Mori, "Flexible scheduling in a machining centre through genetic algorithms," *Comput. Indust. Eng.*, vol. 30, no. 4, pp. 931–940, 1996.

[74] E. Hart and D. Corne, "The state of the art in evolutionary approaches to timetabling and scheduling,", EvoStim Working Group on Evolutionary Scheduling and Timetabling, available through WWW: http://www.dai.ed.ac.uk/daidb/people/homes/emmah/REPORT/draftsc/draftsc.html.

[75] C. R. Reeves, "A genetic algorithm for flowshop scheduling," *Comput. Oper. Res.*, vol. 22, no. 1, pp. 5–13, 1995.

[76] C.-L. Chen, R. V. Neppali, and N. Aljaber, "Genetic algorithms applied to the continuous flow-shop problem," *Comput. Indust. Eng.*, vol. 30, no. 4, pp. 919–929, 1996.

[77] T. Murata, H. Ishibuchi, and H. Tanaka, "Genetic algorithms for flowshop scheduling problems," *Comput. Indust. Eng.*, vol. 30, no. 4, pp. 1061–1071, 1996.

[78] P. R. Drake and I. A. Choudry, "From apes to schedules," *Manuf. Eng.*, vol. 76, no. 1, pp. 35–45, 1997.

[79] M. Braglia and E. Gentili, "An improved genetic algorithm for flowshop scheduling problems," in *Proc. 10th ISPE/IFAC Int. Conf. CAD/CAM, Robotics and Factories of the Future*. Ontario, Canada: OCRI, 1994, pp. 137–142.

[80] T. Murata, H. Ishibuchi, and H. Tanaka, "Multi-objective genetic algorithm and its application to flow-shop scheduling," *Comput. Indust. Eng.*, vol. 30, no. 4, pp. 957–968, 1996.

[81] S. S. Lam, K. W. C. Tang, and X. Cai, "Genetic algorithm with pigeon-hole coding scheme for solving sequencing problems," *Appl. Art. Intell.*, vol. 10, no. 3, pp. 239–256, 1996.

[82] R. Sikora, "A genetic algorithm for integrating lot-sizing and sequencing in scheduling a capacitated flow-line," *Comput. Indust. Eng.*, vol. 30, no. 4, pp. 969–981, 1996.

[83] N. Sannomiya and H. Iima, "Application of a genetic algorithm to scheduling problems in manufacturing processes," in *Proc. 1996 IEEE Int. Conf. Evol. Comput*. Piscataway, NJ: IEEE, 1996, pp. 523–528.

[84] ——, "Genetic algorithm approach to a modified flowshop scheduling problem," in *Proc. Japan/USA Symp. Flexible Automat.*: ASME, 1996, pp. 1229–1234.

[85] C. Y. Lee and J. Y. Choi, "A genetic algorithm for job sequencing problems with distinct due dates and general early-tardy penalty weights," *Comput. Oper. Res.*, vol. 22, no. 8, pp. 857–869, 1995.

[86] ——, "Parallel genetic algorithms for the earliness-tardiness job scheduling problem with general penalty weights," *Comput. Indust. Eng.*, vol. 28, no. 2, pp. 231–243, 1995.

[87] I. Lee, R. Sikora, and M. J. Shaw, "A genetic algorithm-based approach to flexible flow-line scheduling with variable lot sizes," *IEEE Trans. Syst., Man, Cybern.—Part B: Cybern.*, vol. 27, pp. 36–54, Feb. 1997.

[88] B. Gonzalez, M. Torres, and J. A. Moreno, "A hybrid genetic algorithm approach for the 'no-wait' flowshop scheduling problem," in *GALESIA'95: Proc. Conf. Genetic Algorithms in Eng. Syst.: Innovations and Appl*. London, U.K.: IEE, 1995, pp. 59–64. IEE Conf. Publ. 414.

[89] J. W. Herrmann and C.-Y. Lee, "Solving a class scheduling problem with genetic algorithms," *ORSA J. Comput.*, vol. 7, no. 4, pp. 443–452, 1995.

[90] S. Karabati and P. Kouvelis, "Flow line scheduling problem with controllable processing times," *IIE Trans.*, vol. 29, no. 1, pp. 1–14, 1997.

[91] H. Ishibuchi, N. Yamamoto, T. Murata, and H. Tanaka, "Genetic algorithms and neighborhood search algorithms for fuzzy flowshop scheduling problems," *Fuzzy Sets Syst.*, vol. 67, no. 1, pp. 81–100, 1994.

[92] I. Kebbe, H. Yokoi, K. Suzuki, and Y. Kakazu, "Vibrational-potentional method for large scale scheduling problems," in *Proc. Int. Conf. Advances in Prod. Syst., APMS'96*, Okino, Tamura, and Fujii, Eds. Kyoto, Japan: Kyoto Univ., 1996, pp. 585–590.

[93] L. Davis, "Applying adaptive algorithms to epistatic domains," in *Proc. 9th Int. Joint Conf. Art. Intell.*, 1985, pp. 162–164.

[94] S. Katoh, Y. Mutsunori, and T. Ibaraki, "Application of genetic algorithms to single machine scheduling problems under changing environment," in *Proc. Int. Conf. Advances in Prod. Syst., APMS'96*, Okino, Tamura, and Fujii, Eds. Kyoto, Japan: Kyoto Univ., 1996, pp. 543–546.

[95] L. Davis, Ed., *Handbook of Genetic Algorithms*. Princeton, NJ: Van Nostrand Reinhold, 1991.

[96] T. Murata and H. Ishibuchi, "Positive and negative combination effects of crossover and mutation operators in sequencing problems," in *Proc. 1996 IEEE Int. Conf. Evol. Comput*. Piscataway, NJ: IEEE, 1996, pp. 170–175.

[97] S. Fichera, V. Grasso, A. Lombardo, and E. L. Valvo, "Genetic algorithms efficiency in flowshop scheduling," in *Proc. 10th Int. Conf. Appl. Art. Intell. in Eng. AIENG'95*. Southampton, U.K.: Comput. Mech., 1995, pp. 261–270.

[98] D. E. Goldberg and R. Lingle, "Alleles, loci and the TSP," in *Proc. 1st Int. Conf. Genetic Algorithms Appl.*, J. J. Grefenstette, Ed. Hillsdale, NJ: Lawrence Erlbaum, 1985, pp. 254–259.

[99] C.-L. Chen, V. S. Vempati, and N. Aljaber, "An application of genetic algorithms for flow shop problems," *Eur. J. Oper. Res.*, vol. 80, no. 2, pp. 389–396, 1995.

[100] C. Rajendran and D. Chaudhuri, "Heuristic algorithms for continuous flow-shop problems," *Nav. Res. Logist.*, vol. 37, pp. 695–705, 1990.

[101] J. Sridhar and C. Rajendran, "A genetic algorithm for family and job scheduling in a flow-line based manufacturing cell," *Comput. Indust. Eng.*, vol. 27, no. 1–4, pp. 469–472, 1994.

[102] ——, "Scheduling in flowshop and cellular manufacturing systems with multiple objectives—A genetic algorithm approach," *Prod. Planning Contr.*, vol. 7, no. 4, pp. 374–382, 1996.

[103] P. Ross and A. Tuson, "Directing the search of evolutionary and neighborhood-search optimizers for the flowshop sequencing problem with idle-time heuristic," in *Evol. Comput., AISB Workshop*, D. Corne and L. Shapiro, Eds. Berlin, Germany: Springer-Verlag, 1997, pp. 213–225. (Lecture Notes in Comput. Sci., 1305).

[104] D. Whitley, T. Starkweather, and D. Fuquay, "Scheduling problems and travelling salesmen: The genetic edge recombination operator," in *Proc. 3rd Int. Conf. Genetic Algorithms Appl.*, J. Shaffer, Ed. San Mateo, CA: Morgan Kaufmann, 1989, pp. 133–140.

[105] T. Starkweather, S. McDaniel, K. Mathias, D. Whitley, and C. Whitley, "A comparison of genetic sequencing operators," in *Proc. 4th Int. Conf. Genetic Algorithms Appl.*, R. Belew and L. Booker, Eds. San Mateo, CA: Morgan Kaufmann, 1991, pp. 69–76.

[106] T. Asveren and P. Molitor, "New crossover methods for sequencing problems," in *Proc. 4th Int. Conf. PPS from Nature*. Berlin, Germany: Springer-Verlag, 1996, pp. 290–299.

[107] T. Yamada and C. R. Reeves, "Permutation flowshop scheduling by genetic local search," in *GALESIA'97: Genetic Algorithms Eng. Syst.: Innovations and Appl*. Stevenage, U.K.: IEE, 1997, pp. 232–238. IEE Conf. Publ. 446.

[108] M. Yagiura and T. Ibaraki, "The use of dynamic programming in genetic algorithms for permutation problems," *Eur. J. Oper. Res.*, vol. 92, no. 2, pp. 387–401, 1996.

[109] I. M. Oliver, D. J. Smith, and J. R. Holland, "A study of permutation crossover operators of the travelling salesman problem," in *Proc. 2nd Int. Conf. Genetic Algorithms Appl.*, J. J. Grefenstette, Ed. Hillsdale, NJ: Lawrence Erlbaum, 1987, pp. 224–230.

[110] J. J. Grefenstette, "A system for learning control strategies with genetic algorithms," in *Proc. 3rd Int. Conf. Genetic Algorithms Appl.*, J. Shaffer, Ed. San Mateo, CA: Morgan Kaufmann, 1989, pp. 160–168.

[111] R. Pakath and J. S. Zaveri, "Specifying critical inputs in a genetic algorithm-driven decision support system," *Decision Sci.*, vol. 26, no. 6, pp. 749–779, 1995.

[112] E. Tailard, "Benchmarks for basic scheduling problems," *Eur. J. Oper. Res.*, vol. 64, pp. 278–285, 1993.

[113] C. Chiu and Y. Yih, "A learning-based methodology for dynamic scheduling in distributed manufacturing systems," *Int. J. Prod. Res.*, vol. 33, no. 11, pp. 3217–3232, 1995.

[114] H. Aytug, G. H. Koehler, and J. L. Snowdon, "Genetic learning of dynamic scheduling within a simulation environment," *Comput. Oper. Res.*, vol. 21, no. 8, pp. 909–925, 1994.

[115] Y. Yih, "Trace-driven knowledge acquisition (TDKA) for rule-based real-time scheduling systems," *J. Intell. Manuf.*, vol. 1, pp. 217–230, 1990.

[116] A. Jones, L. Rabelo, and Y. Yih, "A hybrid approach for real-time sequencing and scheduling," *Int. J. Comput. Integrated Manuf.*, vol. 8, no. 2, pp. 145–154, 1995.

[117] L. C. Rabelo, A. Jones, and Y. Yih, "Neural networks, simulation, genetic algorithms and machine learning for manufacturing scheduling," in *WCCN'95, World Congr. Neural Networks*. Mahwah, NJ: Lawrence Erlbaum, 1995, pp. 130–135.

[118] A. Jones, F. Riddick, and L. C. Rabelo *et al.*, "Development of a predictive-reactive scheduler using genetic algorithms and simulation-based scheduling software," in *Adv. Manufact. Processes Syst. Technol.—AMPST'96*, Khan *et al.*, Eds. Bury St. Edmunds, UK: Mech. Eng. Publ., 1996, pp. 589–598.

[119] C.-Y. Lee, S. Piramuthu, and Y.-K. Tsai, "Job-shop scheduling with a genetic algorithm and machine learning," *Int. J. Prod. Res.*, vol. 35, no. 4, pp. 1171–1191, 1997.

[120] H. Tamaki, M. Ochi, and M. Araki, "Application of genetics-based machine learning to production scheduling," in *Japan/USA Symp. Flexible Automation*: ASME, 1996, pp. 1221–1224.

[121] Y. Ikkai, M. Inoue, T. Ohkawa, and N. Komoda, "A learning method of scheduling knowledge by genetic algorithms," in *Proc. 1995 IEEE Symp. Emerging Technol. Factory Automation*. Los Alamitos, CA: IEEE, 1995, pp. 641–648.

[122] J. Fang and Y. Xi, "A rolling horizon job shop rescheduling strategy in the dynamic environment," *Int. J. Adv. Manuf. Technol.*, vol. 13, no. 3, pp. 227–232, 1997.

[123] H. M. Cartwright and A. L. Tuson, "Genetic algorithms and flowshop scheduling: Toward the development of a real-time process control system," in *Evol. Comput., AISB Workshop, Selected Papers*, T. C. Fogarty, Ed. Berlin, Germany: Springer-Verlag, 1994, pp. 277–290. (Lecture Notes in Comput. Sci., 865).

[124] C. Bierwirth, H. Kopfer, D. C. Mattfeld, and I. Rixen, "Genetic algorithm based scheduling in a dynamic manufacturing environment," in *Proc. 1995 IEEE Conf. Evol. Comput*. Piscataway, NJ: IEEE, 1995, vol. 1, pp. 439–443.

[125] A. K. Jain and H. A. Elmaraghy, "Production scheduling/rescheduling in flexible manufacturing systems," *Int. J. Prod. Res.*, vol. 35, no. 1, pp. 281–309, 1997.

[126] E .P .K. Tsang, "Scheduling techniques—A comparative study," *BT Technol. J.*, vol. 13, no. 1, pp. 16–29, 1995.

[127] J. Dorn, M. Girsch, G. Skele, and W. Slany, "Comparison of iterative improvement techniques for schedule optimization," *Eur. J. Oper. Res.*, vol. 94, no. 2, pp. 349–361, 1996.

[128] R. E. Korf, "Depth-first iterative deepening," *Art. Intell.*, vol. 27, no. 1, pp. 97–109, 1987.

[129] M. Yagiura and T. Ibaraki, "Meta-heuristics as simple and robust optimization tools," in *Proc. 1996 IEEE Conf. Evol. Comput*. Piscataway, NJ: IEEE, 1996, pp. 541–546.

[130] G. McMahon and D. Hadinoto, "Comparison of heuristic search algorithms for single-machine scheduling problems," in *Proc. AI'93 and AI'94 Workshops Evol. Comput.*, X. Yao, Ed. Berlin, Germany: Springer-Verlag, 1995, pp. 293–303.

[131] C. A. Glass and C. N. Potts, "A comparison of local search methods for flowshop scheduling," *Ann. Oper. Res.*, vol. 63, pp. 489–509, 1996.

[132] M. Horvath, A. Markus, and C. Vancza, "Process planning with genetic algorithms on results of knowledge-based reasoning," *Int. J. Comput. Integr. Manuf.*, vol. 9, no. 2, pp. 145–166, 1996.

[133] J. M. Usher and R. Bowden, "The application of genetic algorithms to operation sequencing for use in computer-aided process planning," *Comput. Indust. Eng.*, vol. 30, no. 4, pp. 999–1013, 1996.

[134] D. Yip-Hoi and P. Dutta, "A genetic algorithm application for sequencing operations in process planning for parallel machining," *IIE Trans.*, vol. 28, no. 1, pp. 55–68, 1996.

[135] N. Takatori, M. Minagawa, and Y. Kakazu, "A GA-based approach to a process palnning problem with geometric constraints," in *ANNIE'94: Proc. Art. Neural Networks in Eng*. New York: ASME Press, 1994, pp. 369–374.

[136] H. N. Kanhawi, S. R. Leclair, and C. L. Philip, "Feature sequencing in the rapid design system using a genetic algorithm," *J. Intell. Manuf.*, vol. 7, no. 1, pp. 55–67, 1996.

[137] B. A. Norman and G. C. Bean, "Operation sequencing and tool assignment for multiple spindle CNC machines," in *Proc. 1997 IEEE Conf. Evol. Comput.*. Piscataway, NJ: IEEE, 1997, pp. 425–429.

[138] B. Awadh, N. Sepehri, and O. Hawaleshka, "A computer-aided process planning model based on genetic algorithms," *Comput. Oper. Res.*, vol. 22, no. 8, pp. 841–856, 1995.

[139] G. Zhou and M. Gen, "Evolutionary computation of multi-criteria production process planning problem," in *Proc. 1997 IEEE Conf. Evol. Comput.*. Piscataway, NJ: IEEE, 1997, pp. 419–424.

[140] N. Singh, *Systems Approach to Computer-Integrated Design and Manufacturing*. Chichester, NY: Wiley, 1996.

[141] P. Husbands, "Distributed co-evolutionary genetic algorithms for multi-criteria and multi-constrained optimisation," in *Evol. Comput., AISB Workshop, Selected Papers*, T. C. Fogarty, Ed. Berlin, Germany: Springer-Verlag, 1994, pp. 150–165. (Lecture Notes in Comput. Sci., 865).

[142] M. McIlhagga, P. Husbands, and R. Ives, "A comparison of optimization techniques for integrated manufacturing planning and scheduling," in *Proc. 4th Int. Conf. PPS from Nature*, Voigt, Ebeling, Rechenberg, and Schwefel, Eds. Berlin, Germany: Springer-Verlag, 1996, pp. 604–613.

[143] R. Collins and D. Jefferson, "Selection in massively parallel genetic algorithms," in *Proc. 4th Int. Conf. Genetic Algorithms*, Belew and Booker, Eds., 1991, pp. 249–256.

[144] R. Bowden and S. F. Bullington, "Development of manufacturing control strategies using unsupervised machine learning," *IIE Trans.*, vol. 28, no. 4, pp. 319–331, 1996.

[145] J. J. Grefenstette, "Strategy acquisition with genetic algorithms," in *Handbook of Genetic Algorithms*, L. Davis, Ed. New York: Van Nostrand Reinhold, 1991.

[146] F. Zhang, Y. F. Zhang, and A. Y. C. Nee, "Using genetic algorithms in process planning for job shop machining," *IEEE Trans. Evol. Comput.*, vol. 1, no. 4, pp. 278–289, 1998.

[147] Y. Hayashi, H. Kim, and K. Nava, "Scenario creation method by genetic algorithms for evaluating future plans," in *Proc. 1996 IEEE Conf. Evol. Comput.*. Piscataway, NJ: IEEE, 1996, pp. 880–885.

[148] S. P. Mitrovanov, *The Scientific Principles of Group Technology*, 1966. (English transl.).

[149] J. L. Burbidge, "Production flow analysis," *Prod. Eng.*, vol. 42, p. 472, 1963.

[150] ——, *The Introduction of Group Technology*. New York: Wiley, 1975.

[151] J. A. Brandon, *Cellular Manufacturing: Integrating Technology and Management*. Somerset, England: Res. Studies Press, 1996.

[152] N. L. Hyer and U. Wemmerlov, "Group technology and productivity," *Harvard Bus. Rev.*, vol. 4, pp. 140–149, 1984.

[153] N. L. Hyer, "Case studies in manufacturing cells: Implications of research," *Proc. Decision Sci. Inst.*, pp. 1407–1409, 1991.

[154] N. L. Hyer and U. Wemmerlov, "Group technology in U.S. manufacturing industry: A survey of current practices," *Int. J. Prod. Res.*, vol. 27, no. 8, pp. 1287–1304, 1989.

[155] U. Wemmerlov and N. L. Hyer, "Cellular manufacturing in the U.S. industry: A survey of users," *Int. J. Prod. Res.*, vol. 27, no. 9, pp. 1511–1530, 1989.

[156] N. Singh, "Design of cellular manufacturing systems: An invited review," *Eur. J. Oper. Res.*, vol. 69, pp. 284–291, 1993.

[157] O. Offodile, A. Mehrez, and J. Grznar, "Cellular manufacturing: A taxonomic review framework," *J. Manuf. Syst.*, vol. 13, no. 3, pp. 196–220, 1994.

[158] N. Morad, "Optimisation of cellular manufacturing systems using genetic algorithms," Ph.D. dissertation, Univ. Sheffield, England, 1997.

[159] J. R. King, "Machine-component grouping in production flow analysis: An approach using rank-order clustering algorithm," *Int. J. Prod. Res.*, vol. 18, no. 1, pp. 213–232, 1980.

[160] H. A. Chan and D. A. Milner, "Direct clustering algorithm for group formation in cellular manufacturing," *J. Manuf. Syst.*, vol. 1, pp. 65–72, 1982.

[161] J. McAuley, "Machine grouping for efficient production," *Prod. Eng.*, vol. 51, no. 2, pp. 53–57, 1972.

[162] M. V. Tatikonda and U Wemmerlov, "Adoption and implementation of group technology classification and coding systems: Insights from seven case studies," *Int. J. Prod. Res.*, vol. 30, pp. 2087–2110, 1992.

[163] R. Rajagopalan and J. L. Balta, "Design of cellular production systems: A graph-partitioning theoretic approach," *Int. J. Prod. Res.*, vol. 13, no. 6, pp. 567–579, 1975.

[164] F. Boctor, "A linear formulation of the machine-part cell formation problem," *Int. J. Prod. Res.*, vol. 29, no. 2, pp. 343–356, 1991.

[165] S. Kaparthi and N. C. Suresh, "Machine-component cell formation in group technology: A neural network approach," *Int. J. Prod. Res.*, vol. 26, no. 6, pp. 1353–1367, 1992.

[166] C.-H. Chu and J. C. Hayya, "A fuzzy-clustering approach to manufacturing cell formation," *Int. J. Prod. Res.*, vol. 29, no. 7, pp. 1475–1487, 1991.

[167] V. Venugopal and T. T. Narendran, "A genetic algorithm approach to the machine-component grouping problem with multiple objectives," *Comput. Indust. Eng.*, vol. 22, no. 4, pp. 269–480, 1992.

[168] Y. Gupta, M. Gupta, A. Kumar, and C. Sundaram, "A genetic algorithm-based approach to cell-composition and layout design problems," *Int. J. Prod. Res.*, vol. 34, no. 2, pp. 447–482, 1996.

[169] ——, "Minimising total intercell and intracell moves in cellular manufacturing: A genetic algorithm approach," *Int. J. Comput. Integr. Manuf.*, vol. 8, no. 2, pp. 92–101, 1995.

[170] R. E. Billo, B. Bidanda, and D. Tate, "A genetic cluster algorithm for the machine-component grouping problem," *J. Intell. Manuf.*, vol. 7, no. 3, pp. 229–243, 1996.

[171] J. A. Joines, C. T. Culbreth, and R. E. King, "Manufacturing cell design: An integer programming model employing genetic algorithms," *IIE Trans.*, vol. 28, no. 1, pp. 69–85, 1996.

[172] C.-T. Su and C.-M. Hsu, "A two-phased genetic algorithm for the cell formation problem," *Int. J. Industr. Eng.*, vol. 3, no. 2, pp. 114–125, 1996.

[173] N. Morad and A. M. S. Zalzala, "A genetic-based approach to the formation of manufacturing cells and batch scheduling," in *Proc. 1996 IEEE Int. Conf. Evol. Comput.* Piscataway, NJ: IEEE, 1996, pp. 485–490.

[174] H. Pierreval and M.-F. Plaquin, "A genetic algorithm approach to group machines into manufacturing cells," in *Proc 4th Int. Conf CIM and Automat. Technol.* Los Alamitos, CA: IEEE, 1994, pp. 267–271.

[175] G. A. Süer, "Evolutionary programming for designing manufacturing cells," in *Proc. 1996 IEEE Int. Conf. Evol. Comput.* Piscataway, NJ: IEEE, 1996, pp. 379–384.

[176] C. Dimopoulos and A. M. S. Zalzala, "Optimization of cell configuration and comparisons using evolutionary computation approaches," in *Proc. 1998 IEEE Int. Conf. Evol. Comput.* Piscataway, NJ: IEEE, 1998, pp. 148–153.

[177] J. L. Paris and H. Pierreval, "Manufacturing cell formation using distributed evolutionary algorithms," in *Proc. 12th Int. Conf. CAD/CAM, Robot. and Factories of the Future.* London, U.K.: Middlesex Univ. Press, 1996, pp. 369–374.

[178] H. Hwang and J.-U. Sun, "A genetic algorithm-based heuristic for the GT cell formation problem," *Comput. Indust. Eng.*, vol. 30, no. 4, pp. 941–955, 1996.

[179] L. Zhao, Y. Tsujimura, and M. Gen, "Genetic algorithm for fuzzy clustering," in *Proc. 1996 IEEE Int. Conf. Evol. Comput.* Piscataway, NJ: IEEE, 1996, pp. 716–719.

[180] A. Kusiak and S. S. Heragu, "The facility layout problem," *Eur. J. Oper. Res.*, vol. 29, pp. 229–251, 1987.

[181] S. Sahni and T. Gonzalez, "P-complete approximation problems," *J. Assoc. Comput. Mach.*, vol. 23, pp. 555–565, 1976.

[182] T. D. Mavridou and P. M. Pardalos, "Simulated annealing and genetic algorithms for the facility layout problem: A survey," *Comput. Optimis. Appl.*, vol. 7, no. 1, pp. 111–126, 1997.

[183] J. P. Cohoon, S. U. Hedge, W. N. Martin, and D. S. Richards, "Distributed genetic algorithms for the floorplan design problem," *IEEE Trans. Computer-Aided Design*, vol. 10, no. 4, pp. 483–492, 1992.

[184] K. Y. Tam, "Genetic algorithms, function optimisation and facility layout design," *Eur. J. Oper. Res.*, vol. 63, no. 2, pp. 322–346, 1992.

[185] R. H. J. M. Otten, "Automatic floorplan design," in *Proc. 19th ACM–IEEE Design Automat. Conf.*, 1982, pp. 261–267.

[186] K. Kado, P. Ross, and D. Corne, "A study of genetic algorithms for facility layout problems," in *Proc. 6th Int. Conf. Genetic Algorithms.* San Mateo, CA: Morgan Kaufmann, 1995, pp. 499–505.

[187] J. Garces-Perez, D. A. Schoenefeld, and R. L. Wainwright, "Solving facility layout problems using genetic programming," in *Genetic Programming 1996: Proc. 1st Annu. Conf.*, Koza, Goldberg, Fogel, and Riolo, Eds. Cambridge, MA: MIT Press, 1996, pp. 182–190.

[188] R. Cheng, M. Gen, and T. Tozawa, "Genetic search for facility layout design under intreflows uncertainty," in *Proc. 1995 IEEE Conf. Evol. Comput.*. Piscataway, NJ: IEEE, 1995, vol. 1, pp. 400–405.

[189] D. M. Tate and A. E. Smith, "Unequal-area facility layout by genetic search," *IIE Trans.*, vol. 7, no. 4, pp. 465–472, 1995.

[190] B. A. Norman and A. E. Smith, "Random-keys genetic algorithm with adaptive penalty function for optimization of constrained facility layout problems," in *Proc. 1997 IEEE Int. Conf. Evol. Comput.* Piscataway, NJ: IEEE, 1997, pp. 407–411.

[191] G. Suresh, V. V. Vivod, and S. Sahu, "A genetic algorithm for facility layout," *Int. J. Prod. Res.*, vol. 33, no. 12, pp. 3411–3423, 1995.

[192] M. Kazerooni, L. H. S. Luong, K. Abhary, F. T. S. Chan, and F. Pun, "An integrated method for cell layout problem using genetic algorithms," in *Proc. 12th Int. Conf. CAD/CAM, Robotics and Factories of the Future.* London, England: Middlesex Univ. Press, 1996, pp. 752–762.

[193] P. Banerjee, Y. Zhou, and B. Montreuil, "Genetically assisted optimization of cell layout and material flow path skeleton," *IIE Trans.*, vol. 29, no. 4, pp. 277–291, 1997.

[194] D. G. Conway and M. A. Venkataramanan, "Genetic search and the dynamic facility layout problem," *Comput. Oper. Res.*, vol. 21, no. 8, pp. 955–960, 1994.

[195] M. Bazargan-Lari and H. Kaebernick, "An approach to the machine layout problem in a cellular manufacturing environment," *Prod. Planning Contr.*, vol. 8, no. 1, pp. 41–55, 1997.

[196] M. Braglia and A. Sternieri, "A genetic algorithm for facility layout optimization in a flowline cellular manufacturing system," in *Proc. Int. ICSC Symp. Intell. Indust. Automat. and Soft Computing*, Anderson and Warwick, Eds. Millet, Canada: Int. Comput. Sci. Conventions, 1996, pp. A21–A26.

[197] M. Braglia and L. Zavonella *et al.*, "Backtracking of jobs and single row machine layout problems in flexible cellular manufacturing system," in *Adv. Manuf. Processes Syst. Technol., AMPST'96*, Khan *et al.*, Eds. Bury St. Edmunds, U.K.: Mech. Eng. Publ., 1996, pp. 17–25.

[198] M. Braglia, "Optimisation of a simulated annealing-based heuristic for a single row machine layout problem by genetic algorithm," *Int. Trans. Oper. Res.*, vol. 3, no. 1, pp. 37–49, 1996.

[199] R. Cheng, M. Gen, and T. Tosawa, "Genetic algorithms for designing loop layout manufacturing systems," *Comput. Indust. Eng.*, vol. 31, no. 3/4, pp. 587–591, 1996.

[200] M. Gen, K. Ida, and C. Cheng, "Multirow machine layout problems in fuzzy environment using genetic algorithms," *Comput. Indust. Eng.*, vol. 29, no. 1-4, pp. 519–523, 1995.

[201] A. Bolte and U. W. Thoneman, "Optimising simulated annealing schedules with genetic algorithms," *European J. Oper. Res.*, vol. 92, no. 2, pp. 402–416, 1996.

[202] M. F. Sebaaly and H. Fujimoto, "A new approach for constrained GA-search: Assembly sequence planning—A case study," in *Proc. Int. ICSC Symp. Intell. Indust. Automat. and Soft Comput.*, Anderson and Warwick, Eds. Millet, Canada: Int. Comput. Sci. Conventions, 1996, pp. B138–B144.

[203] ——, "Integrated planning and scheduling for job-shop assembly based on genetic algorithms," in *Proc. Int. Conf. Adv. Prod. Syst., APMS'96*, Okino, Tamura, and Fujii, Eds. Kyoto, Japan: Kyoto Univ., 1996, pp. 557–562.

[204] ——, "A genetic planner for assembly automation," in *Proc. 1996 IEEE Int. Conf. Evol. Comput.*. Piscataway, NJ: IEEE, 1996, pp. 401–406.

[205] ——, "Assembly sequence planning by GA search: A novel approach," in *Proc. Japan/USA Symp. Flexible Automat.*: ASME, 1996, pp. 1235–1240.

[206] R. Groppeti and R. Muscia, “Genetic algorithms for optimal assembly planning,” in *Proc. 1st World Congr. Intell. Manuf. Processes Syst.*. San Juan, Puerto Rico: Univ. of Puerto Rico, 1995, pp. 319–333.

[207] Y. K. Kim, C. J. Hyun, and Y. Kim, “Sequencing in mixed-model assembly lines: A genetic algorithm approach,” *Comput. Oper. Res.*, vol. 23, no. 12, pp. 1131–1145, 1996.

[208] Y.-Y. Leu, L. A. Matheson, and L. P. Rees, “Sequencing mixed-model assembly lines with genetic algorithms,” *Comput. Indust. Eng.*, vol. 30, no. 4, pp. 1027–1036, 1996.

[209] Y. Monden, *Toyota Production System*. Norcross, CA: Indust. Eng. and Manage. Press, 1993.

[210] G. Suresh, V. V. Vivod, and S. Sahu, “A genetic algorithm for assembly line balancing,” *Prod. Planning Contr.*, vol. 7, no. 1, pp. 38–46, 1996.

[211] J. Rubinovitz and G. Levitin, “Genetic algorithm for assembly line balancing,” *Int. J. Prod. Econ.*, vol. 41, no. 1–3, pp. 343–354, 1995.

[212] Y. Tsujimura, M. Gen, and E. Kubota, “Solving fuzzy assembly-line balancing problems with genetic algorithms,” *Comput. Indust. Eng.*, vol. 29, no. 1–4, pp. 543–547, 1995.

[213] A. A. Bulgak, P. D. Diwan, and B. Inozu, “Buffer size optimisation in asynchronous assembly systems using genetic algorithms,” *Comput. Indust. Eng.*, vol. 28, no. 2, pp. 309–322, 1995.

[214] A. Roach and R. Nagi, “A hybrid GA-SA algorithm for just-in-time scheduling of multi-level assemblies,” *Comput. Indust. Eng.*, vol. 30, no. 4, pp. 1047–1060, 1996.

[215] G. List, “Heuristic methods in a flexible assembly system,” in *Intell. Manuf. Syst. 1994, IMS’94*. Oxford, U.K.: Pergamon, 1994, pp. 251–254.

[216] T. Watanabe, Y. Hashimoto, I. Nishikawa, and H. Tokumaru, “Line balancing using a genetic evolution model,” *Contr. Eng. Practice*, vol. 3, no. 1, pp. 69–76, 1995.

[217] T. Alander and J. Lampinen, “On implementing CAD systems based on existing simulators and optimisation by genetic algorithms,” in *Proc. 3rd Int. Conf. Genetic Algorithms, Optimisation Problems, Fuzzy Logic, Neural Networks and Rough Sets, MENDEL’97*, 1997, pp. 7–11.

[218] Y. J. Cao and Q. H. Wu, “Mechanical design optimisation by mixed-variable evolutionary programming,” in *Proc. 1997 IEEE Int. Conf. Evol. Comp*. Piscataway, NJ: IEEE, 1997, pp. 443–446.

[219] K. Rasheed, H. Hirsch, and A. Gelsey, “A genetic algorithm for continuous design search space,” *Art. Intell. Eng.*, vol. 11, pp. 295–305, 1997.

[220] C. A. Coello and A. D. Christiansen, “An approach to multi-objective optimization using genetic algorithms,” in *ANNIE’95, Proc. Art. Neural Networks in Eng.*, 1995, pp. 410–416.

[221] S. S. Altus, I. M. Kroo, and P. J. Cage, “A genetic algorithm for scheduling and decomposition of multidisciplinary design problems,” *Trans. ASME*, vol. 118, no. 4, pp. 486–489, 1996.

[222] A. C. Thornton and A. L. Johnson, “CADET: A software support toll for constraint processes in embodiment design,” *Res. Eng. Des.*, vol. 8, no. 1, pp. 1–13, 1996.

[223] S. E. Carlson, “Genetic algorithm attributes for component selection,” *Res. Eng. Des.*, vol. 8, no. 1, pp. 33–51, 1996.

[224] M. Iannuzzi and E. Sandgren, “Optimal tolerancing: The link between design and manufacturing productivity,” in *Proc. 6th Int. Conf. Design Theory and Methodol*. New York: ASME, 1994, vol. DE-68, pp. 29–42.

[225] J. G. Ziegler and N. B. Nichols, “Optimum settings for automatic controllers,” *Trans. ASME*, vol. 64, pp. 759–768, 1942.

[226] A. H. Jones and P. B. D. M. Oliveira, “Genetic auto-tuning of PID controllers,” in *GALESIA’95: Proc. Conf. Genetic Algorithms in Eng. Syst.: Innovations and Appl*. London, U.K.: IEE, 1995, pp. 141–145. IEE Conf. Publ. 411.

[227] P. B. D. M. Oliveira and A. H. Jones, “Robust co-evolutionary design of SISO Smith predictor PID controllers,” in *GALESIA’97: Proc. Conf. Genetic Algorithms in Eng. Syst.: Innovations and Appl*. Stevenage, England: IEE, 1997, pp. 504–509. IEE Conf. Publ. 446.

[228] A. H. Jones and B. Porter, “On-line genetic tuning of digital PID controllers,” in *Proc. IASTED Int. Conf.*, M. H. Hamza, Ed. Anaheim, CA: IASTED, 1994, pp. 32–36.

[229] A. H. Jones, N. Ajlouni, S. B. Kenway, and P. B. D. M. Oliveira, “Genetic design of robust PID controllers to deal with prescribed plant uncertainties through a process of competitive co-evolution,” in *Proc. 1996 IEEE Int. Symp. Intell. Contr.*. New York: IEEE, 1996, pp. 372–377.

[230] R. Krohling, “Design of PID controller for disturbance rejection: A genetic optimisation approach,” in *GALESIA’97: Proc. Conf. Genetic Algorithms in Eng. Syst.: Innovations and Appl*. Stevenage, England: IEE, 1997, pp. 498–503. IEE Conf. Publ. 446.

[231] C. Vlachos, J. T. Evans, and D. Williams, “PI controller tuning for multivariable processes using genetic algorithms,” in *GALESIA’97: Proc. Conf. Genetic Algorithms in Eng. Syst.: Innovations and Appl*. Stevenage, U.K.: IEE, 1997, pp. 43–49. IEE Conf. Publ. 446.

[232] M. Salami and G. Cain, “An adaptive PID controller based on genetic algorithm processor,” in *GALESIA’95: Proc. Conf. Genetic Algorithms in Eng. Syst.: Innovations and Appl*. Stevenage, U.K.: IEE, 1995, pp. 88–93. IEE Conf. Publ..

[233] M. Salami, “A multiple genetic algorithm processor for a PID controller system,” in *Proc. 2nd Int. Conf. Genetic Algorithms, Optimisation Problems, Fuzzy Logic, Neural Networks and Rough Sets: MENDEL’95*, 1995, pp. 133–138.

[234] X. M. Qi and T. C. Chin, “Genetic algorithms-based fuzzy controller for high order systems,” *Fuzzy Sets Syst.*, vol. 91, pp. 279–284, 1997.

[235] J. Kim and B. P. Ziegler, “Hierarchical distributed genetic algorithms: A fuzzy logic controller application,” *IEEE Expert*, vol. 11, no. 3, pp. 76–84, 1996.

[236] Y. S. Tarng, Z. M. Yeh, and C. Y. Nian, “Genetic synthesis of fuzzy logic controllers in turning,” *Fuzzy Sets Syst.*, vol. 83, no. 3, pp. 301–310, 1996.

[237] H. Polheim and P. Marenback, “Generation of structured process models using genetic programming,” in *Evol. Comput., AISB Workshop, Selected Papers*, T. C. Fogarty, Ed. Berlin, Germany: Springer-Verlag, 1996, pp. 102–109. (Lecture Notes in Computer Science, 1143).

[238] B. M. McKay, M. J. Willis, H. G. Hiden, G. A. Montague, and G. W. Barton, “Identification of industrial processes using genetic programming,” in *Proc. Conf. Identification in Eng. Syst.*, Friswell and Mottershead, Eds. Swansea, U.K.: Univ. of Wales, 1996, pp. 510–519.

[239] C. R. Reeves, P. Dai, and K. J. Burham, “A hybrid genetic algorithm for system identification,” in *Proc. Int. ICSC Symp. Intell. Indust. Automat. Soft Comput.*, Anderson and Warwick, Eds. Millet, Canada: Int. Comput. Sci. Conventions, 1996, pp. B278–B283.

[240] P. Chen, T. Toyota, and M. Nasu, “Self-organization method of symptom parameters for for failure diagnosis by genetic algorithms,” in *Proc. 1996 IEEE Indust. Electron. Conf., IECON*. Piscataway, NJ: IEEE, 1996, pp. 829–835.

[241] Z. R. Petrovic and S. L. Ivanovic, “Integration of neural networks and genetic algorithms: An example of machine noise diagnosis,” in *Proc. 1st World Congr. Intell. Manuf. Processes and Syst*. San Juan, PR: Univ. of Puerto Rico, 1995, pp. 962–971.

[242] C.-R. Guzman and M. A. Kramer, “Remote diagnosis and monitoring of complex industrial systems using a genetic algorithm approach,” in *Proc. IEEE Int. Symp. Indust. Electron*. Piscataway, NJ: IEEE, 1994, pp. 363–376.

[243] H. Kim, N. Koichi, and M. Gen, “A method for maintenance scheduling using GA combined with SA,” *Comput. Indust. Eng.*, vol. 27, no. 1–4, pp. 477–480, 1994.

[244] N. Viswanadham, S. M. Sharma, and M. Taneja, “Inspection allocation in manufacturing systems using stochastic search techniques,” *IEEE Trans. Syst., Man, Cybern.—Part A: Syst. and Humans*, vol. 26, pp. 222–230, Mar. 1996.

[245] S. Patro and W. J. Kolarik, “Neural networks and evolutionary computation for real-time quality control of complex processes,” in *Proc. 1997 IEEE Annu. Rel. Maintain. Symp*. Piscataway, NJ: IEEE, 1997, pp. 327–332.

[246] C. G. Lu, D. Morton, Z. Wang, P. Myler, and M. H. Wu, “A genetic algorithm solution of inspection path planning system for multiple tasks inspection on coordinate measuring machine (CMM),” in *GALESIA’95: Proc. Conf. Genetic Algorithms in Eng. Syst.: Innovations and Appl.*. London, U.K.: IEE, 1995, pp. 436–441. IEE Conf. Publ. 414.

[247] K. L. Mak and Y. S. Wong, “Design of integrated production production-inventory-distibution systems using genetic algorithms,” in *GALESIA’95: Proc. Conf. Genetic Algorithms in Eng. Syst.: Innovations and Appl*. London, U.K.: IEE, 1995, pp. 454–460. IEE Conf. Publ. 414.

[248] S. M. Disney, M. M. Naim, and D. R. Towill, “Development of a fitness measure for an inventory and production control system,” in *GALESIA’97: Proc. Conf. Genetic Algorithms in Eng. Syst.: Innovations and Appl*. Stevenage, U.K.: IEE, 1997, pp. 351–355. IEE Conf. Publ. 446.

[249] D. Gong, G. Yamazaki, and M. Gen, “Evolutionary program for optimal design of material distribution system,” in *Proc. 1996 IEEE Int. Conf. Evol. Comput.*. Piscataway, NJ: IEEE, 1996, pp. 139–143.

[250] D. J. Stockton and L. Quinn, “Aggregate production planning using genetic algorithms,” *Proc. Inst. Mech. Eng., Part B: J. Eng. Manuf.*, vol. 209, no. B3, pp. 201–209, 1995.

[251] D. Wang and S.-C. Fang, "A genetics-based approach for aggregated production planning in a fuzzy environment," *IEEE Trans. Syst., Man, Cybern.—Part A: Systems and Humans*, vol. 27, no. 5, pp. 636–645, 1997.

[252] W. Feng, G. B. Burns, and D. K. Harrison, "Using genetic algorithms bounded by dynamic linear constraints for marketing/production joint decision making," in *GALESIA'97: Proc. Conf. Genetic Algorithms in Eng. Syst.: Innovations and Appl*. Stevenage, U.K.: IEE, 1997, pp. 339–344. IEE Conf. Publ. 446.

[253] A. G. Garavelli, O. G. Okogbaa, and N. Violante, "Global manufacturing systems: A model supported by genetic algorithms to optimise production planning," *Comput. Indust. Eng.*, vol. 31, no. 1/2, pp. 193–196, 1996.

[254] X. Jinxing, "An application of genetic algorithms for general dynamic lotsizing problems," in *GALESIA'97: Proc. Conf. Genetic Algorithms in Eng. Syst.: Innovations and Appl.*, Stevenage, U.K.: IEE, 1997, pp. 82–87. IEE Conf. Publ. 446.

[255] Q. Xia and S. Macchietto, "Design and synthesis of batch plants—MINLP solution based on a stochastic method," *Comput. Chem. Eng.*, vol. 21, pp. S697–S702, 1997. suppl. issue.

[256] D. G. Kim, D. Corne, and P. Ross, "Industrial plant pipe-route optimisation with genetic algorithms," in *Proc. 4th Int.Conf. PPS from Nature*, Voigt, Ebeling, Rechenberg, and Schwefel, Eds. Berlin, Germany: Springer-Verlag, 1996, pp. 1012–1021.

[257] R. O. Bowden, J. D. Hall, and J. M. Usher, "Integration of evolutionary programming and simulation to optimisse a pull production system," *Comput. Indust. Eng.*, vol. 31, no. 1/2, pp. 217–220, 1996.

[258] H. A. Rao and P. Gu, "Entropic measure to determine reconfiguration using integrated systems design," in *Proc. 1995 IEEE Int. Conf. Syst., Man, Cybern*. New York: IEEE, 1995, pp. 518–523.

[259] N. Kubota, T. Fukuda, and K. Shimojima, "Virus-evolutionary algorithm for a self-organising manufacturing system," *Comput. Indust. Eng.*, vol. 30, no. 4, pp. 1015–1026, 1996.

[260] Y. Kawauchi, M. Inaba, and T. Fukuda, "Genetic evolution and self-organisation of cellular robotic system," *JSME Int. J., Ser. C: Dyn., Contr., Robot., Design and Manuf.*, vol. 38, no. 3, pp. 501–509, 1995.

[261] L. Zhao, Y. Tsujimura, and M. Gen, "Genetic algorithm for robot selection and workstation assignment problem," *Comput. Indust. Eng.*, vol. 31, no. 3/4, pp. 599–602, 1996.

[262] M. McIlhaga, "Solving generic scheduling problems with a distributed genetic algorithm," in *Evol. Comput., AISB Workshop*, D. Corne and L. Shapiro, Eds. Berlin, Germany: Springer-Verlag, 1997, pp. 199–212. (Lecture Notes in Comput. Sci., 1305).

[263] Y. Hamaguchi, H. Inoue, T. Jinguji, H. Yoshida, and M. Shiozawa, "Transportation scheduling system based on evolution algorithm and super parallel computer," in *Steps Forward: Proc. 2nd World Congr. Intell. Transport Syst.'95*. Tokyo, Japan: Vehicle, Road & Traffic Int. Soc., 1995, pp. 2027–2030.

[264] M. Furukawa, M. Watanabe, A. Mizoe, T. Watanabe, and Y. Kakazu, "Evolutionary computation applied to the logistic CIM system," in *Proc. Int. Conf. Advances in Product. Syst., APMS'96*, Okino, Tamura, and Fujii, Eds. Kyoto, Japan: Kyoto Univ., 1996, pp. 319–324.

[265] W. Mergenthaler, W. Stadler, H. Wilbertz, and N. Zimmer, "Optimizing automotive manufacturing sequences using simulated annealing and genetic algorithms," *Contr. Eng. Practice*, vol. 3, no. 4, pp. 569–573, 1995.

[266] S. Sette, L. Boullart, and L. V. Langenhore, "Optimising a production process by a neural network/genetic algorithm approach," *Eng. Appl. Art. Intell.*, vol. 9, no. 6, pp. 681–689, 1996.

[267] B. A. Peters and M. Rajasekharan, "A genetic algorithm for determining facility design and configuration of single-stage flexible electronic assembly systems," *J. Manuf. Syst.*, vol. 15, no. 5, pp. 316–324, 1996.

[268] P. G. Bachhouse, A. F. Fotheringham, and G. Allan, "A comparison of a genetic algorithm with an experimental design technique in the optimisation of a production process," *J. Oper. Res. Soc.*, vol. 48, no. 3, pp. 247–254, 1997.

[269] F. Corno, P. Prinetto, M. Rebaudengo, M. S. Reorda, and S. Bisotto, "Optimising area loss in flat glass cutting," in *GALESIA'97: Proc. Conf. Genetic Algorithms in Eng. Syst.: Innovations and Appl.*, Stevenage, England: IEE, 1997, pp. 450–455. IEE Conf. Publ. 446.

Christos Dimopoulos was born in Athens, Greece in 1973. He received the B.Sc. degree in automation from the Technological Education Institute (T.E.I.) of Piraeus, Piraeus, Greece, in 1995 and the M.Sc. degree in control systems from the University of Sheffield, Sheffield, U.K., in 1997 where he is working toward the Ph.D. degree.

His research interests include genetic programming applications, with particular focus on manufacturing optimization.

Ali M. S. Zalzala (S'82–M'90) was born in Cairo, Egypt, in 1965. He received the B.Eng. degree in electrical engineering from Kuwait University in 1987, and the Ph.D. degree in control engineering from Sheffield University, Sheffield, U.K., in 1990.

He was a Research Associate with Queen's University of Belfast until 1992, and a Lecturer with Sheffield University until 1999. He currently holds the post of a Senior Lecturer at Heriot-Watt University, Edinburgh, Scotland, U.K. His research interests include the application of soft computing techniques in intelligent systems, in particular robotics, manufacturing optimization, and medical applications, and has over 100 publications. His research funding over the years has included grants from EPSRC (U.K.), the European Commission, and industry.

Dr. Zalzala is a Chartered Engineer, a member of IEE (U.K.), chaired the IEE Professional Group on Intelligent Control Systems, and chairs and acts as an officer on the Evolutionary Programming Society Board. He is General Chairman of the 2000 Congress on Evolutionary Computation, held in San Diego, CA, July 2000.