J Intell Manuf (2012) 23:615–627
DOI 10.1007/s10845-010-0400-9

Multi-objective fuzzy assembly line balancing using genetic algorithms

P. Th. Zacharia · Andreas C. Nearchou

Received: 8 October 2009 / Accepted: 27 March 2010 / Published online: 11 April 2010

Abstract This paper presents a fuzzy extension of the simple assembly line balancing problem of type 2 (SALBP-2) with fuzzy job processing times since uncertainty, variability, and imprecision are often occurred in real-world production systems. The jobs processing times are formulated by triangular fuzzy membership functions. The total fuzzy cost function is formulated as the weighted-sum of two bi-criteria fuzzy objectives: (a) Minimizing the fuzzy cycle time and the fuzzy smoothness index of the workload of the line. (b) Minimizing the fuzzy cycle time of the line and the fuzzy balance delay time of the workstations. A new multi-objective genetic algorithm is applied to solve the problem whose performance is studied and discussed over known test problems taken from the open literature.

Keywords Assembly line balancing · Genetic algorithms · Multi-objective optimization · Fuzzy logic · Fuzzy numbers

Introduction

Today's highly competitive business environment establishes the requirement on manufacturers to effectively optimize the design of manufacturing systems in the minimum possible time. In this context, the design of real-world manufacturing systems becomes more and more important. Particularly, the design of an efficient assembly line has a considerable industrial importance (Baudin 2002). The assembly line balancing problem (ALBP) is a decision problem arising when an assembly line has to be (re)-configured and consists of determining the optimal partitioning of the assembly work among the workstations in accordance with some objectives (Baybars 1986; Scholl 1999). The decisions taken to solve ALBPs in modern flow-line production systems not only affect the final cost of the products, but also affect the variety of the products manufactured, their final quality, as well as, the time-to-market response. The latter index is strongly depended on the production cycle of the assembly line and constitutes one of the most interesting performance indices in ALBPs.

ALBP is classified into simple ALBP (SALBP), and generalized ALBP (Baybars 1986; Scholl 1999; Scholl and Becker 2006). The latter contains characteristics not contained in SALBP such as operating costs objectives, paralleling of stations, mixed-model production, etc. Two formulation types are commonly used with SALBP: SALBP-1 which attempts to minimize the number of stations for a given fixed cycle time, and SALBP-2 which attempts to minimize the cycle time of the line for a given number of stations. The former type is used when a new assembly line has to be implemented and installed, while the latter type is used in an existing assembly line when changes in the production process and manufacturing requirements occur. Any variant of SALBP is of combinatorial nature and belongs to the *NP*-hard class of combinatorial optimization problems (Scholl 1999). Therefore, exact algorithms can hardly be designed to solve large sizes of any variant of SALBP and consequently the right way to proceed is through the use of heuristics techniques.

Recently, many researchers turned their attention to the use of meta-heuristics for the solution of SALBP. The most notable of this group of algorithms are genetic algo-

P. Th. Zacharia
Department of Mechanical Engineering & Aeronautics,
University of Patras, 26 500 Rio, Greece
e-mail: zacharia@mech.upatras.gr

A. C. Nearchou (✉)
Department of Business Administration, University of Patras,
26 500 Rio, Greece
e-mail: nearchou@upatras.gr

rithms (GAs) (Holland 1975), simulated annealing (SA) (Kirkpatrick et al. 1983), and tabu-search (TS) (Glover 1989, 1990). Anderson and Ferris (1994) developed a GA with an objective function that sums up the maximal station time and put a penalty term for precedence violations. Watanabe et al. (1995) showed experimentally that a GA can obtain in reasonably computing time quasi-optimum solutions for large size ALBPs that cannot be solved by ordinary methods. Kim et al. (1996) addressed the multi-objective SALBP with additional objectives such as the maximization of workload smoothness and the maximization of work relatedness. Heinrici (1994) developed a TS procedure for the solution of SALBP-2, and compared its performance to that of a SA algorithm. Scholl and Voß (1996) and Chiang (1998) have also developed efficient TS procedures for SALBP-2. Sabuncuoglu et al. (2000) addressed SALBP-1 via a modified GA with a special chromosome structure which is partitioned dynamically through the evolution process and elitism implemented using concepts borrowed from SA. Baykasoglu (2006) proposed a multiple objective SA algorithm for simple line and U-type ALBPs with the aim of maximizing both the smoothness index and line performance. Nearchou (2008) tackled the bi-criteria SALBP-2 using a new population heuristic based on the differential evolution model. Multiple experiments on known benchmarks ALBPs showed that this approach is superior to existing multi-objective GAs in terms of quality of solutions. Recently, Zhang and Gen (2009) proposed a multi-objective GA for the mixed-model ALBP considering demand ratio-based cycle time. Ozcan and Toklu (2009) presented a hybrid improvement heuristic approach to simple straight and U-type ALBPs based on the ideas of adaptive learning and SA. Important reviews about ALBPs can be found in (Erel and Sarin 1998; Rekiek et al. 2001; Scholl and Becker 2006; Tasan and Tunali 2008).

However, since data in real-world problems are often afflicted with uncertainty, imprecision and vagueness due to both machine and human factors, they can only be estimated as within uncertainty. In an attempt to treat imprecise data, fuzzy numbers are introduced to represent the processing time of each job, where the membership function of a fuzzy data represents the grade of satisfaction of a decision maker.

Compared to the deterministic ALBPs, few research works have been done so far for the fuzzy line balancing problems (Becker and Scholl 2006; Tasan and Tunali 2008). Moreover, there is a lack in the literature for population heuristics for solving the multi-objective fuzzy SALBP-2. The works of Tsujimura et al. (1995) and Gen et al. (1996) were the first that dealt with fuzzy SALBP-1 via GAs. Brudaru and Valmar (2004) also proposed a rather time lengthy hybrid GA which combines a branch and bound method with a GA for solving fuzzy SALBP-1.

To the best of our knowledge, there has been no previous research concerning fuzzy SALBP-2. Bearing in mind that the data obtained from more realistic situations are imprecise and uncertain, the consideration of fuzziness for the solution of SALBP-2 is of immense interest. Aiming to fill this gap, this paper introduces a new multi-objective GA (MOGA) for solving the fuzzy SALBP-2. In the following we will refer to the proposed approach as f-MOGA (stands for fuzzy MOGA). The fuzzy processing time for each job is represented by triangular fuzzy membership functions. The fuzzy fitness function of each individual solution is formulated as the weighted-sum of multiple fuzzy objectives functions. Three optimization criteria are considered to be minimized: the fuzzy cycle time of the line (as the main optimization criterion), the fuzzy balance delay time, and the fuzzy smoothness index of the workload in the line.

The rest of the paper is organized as follows: Section "Fuzzy assembly line balancing model-2" formulates the fuzzy SALBP-2 and presents the arithmetics and fuzzy ranking numbers. Section "The proposed solution model for the mo fuzzy salbp-2" analyses and describes the basic components of f-MOGA for the solution of the fuzzy SALBP-2. Computational results concerning the performance of f-MOGA under the influence of various formulations of the fitness function are provided in Section "Numerical results and discussion"; while conclusions and directions for future work are pointed out and discussed in Section "Conclusions".

Fuzzy assembly line balancing model-2

Problem formulation

The fuzzy SALBP can be stated as follows: m workstations are arranged along an assembly line. Manufacturing a single product on the line requires the partitioning of the total work into a set $V = \{1, \ldots, n\}$ of n elementary operations called tasks. Each task j is performed on exactly one station and requires a fuzzy processing time $\tilde{t}_j$. Let S_z $(z = 1, \ldots, m)$ be the station load of station z (i.e. the set of tasks assigned to z), with a cumulated fuzzy task time $\widetilde{tS}_z = \sum_{j \in S_z} \tilde{t}_j$ $(z = 1, \ldots, m)$. The tasks are partially ordered by precedence relations defining a directed acyclic graph (DAG) $G = (V, E)$; with V being the set of the nodes denoting the tasks in G and E the set of the edges representing the precedence constraints among these tasks. The assembly line is associated with a fuzzy cycle time $\tilde{c}$ denoting the maximum processing time available for each station. The fuzzy balance efficiency $\widetilde{BE}$ is therefore defined as:

$$\widetilde{BE} = \frac{\widetilde{tS}_z}{m \times \tilde{c}} \tag{1}$$

In this work, the bi-criteria fuzzy SALBP-2 is considered with main objective to minimize the fuzzy cycle time $\tilde{c}$ for a given fixed number of stations m and secondary objective to minimize:

1. The fuzzy smoothness index $\left(\widetilde{SX}\right)$ measuring the equality of the distributed work among the stations. The lower the value of $\widetilde{SX}$ the smoother the line, resulting in reduced in-process inventory. An $\widetilde{SX}$ equal to zero indicates a perfect balance of the workload among the stations.

$$\widetilde{SX} = \sqrt{\sum_{z=1}^{m} \left(\tilde{c} - \tilde{t}S_z\right)^2} \tag{2}$$

2. The fuzzy balance delay time $\left(\widetilde{BD}\right)$ of the line. $\widetilde{BD}$ reflects the unused capacity of the line, i.e. the summation of the idle times of all the stations.

$$\widetilde{BD} = \sum_{z=1}^{m} \left(\tilde{c} - \tilde{t}S_z\right) \tag{3}$$

The multi-objective fuzzy SALBP-2

In multi-objective (MO) fuzzy SALBP-2 we ideally seek for a feasible solution that simultaneously optimizes $\tilde{c}$, as well as, $\widetilde{SX}$ and $\widetilde{BD}$. Since this is almost impossible for any MO problem (Bäck 1996), what we really attempt to do is to optimize each individual objective to the greatest possible extend. The problems considered in this study can be formulated as in the following:

1. MO SALBP-2 version 1:

$$\text{Minimize } F_2 = w_1 \cdot \tilde{c} + w_2 \cdot \widetilde{SX} \tag{4}$$

subject to a partition of the set $V = \{1, \ldots, n\}$ into m disjoint subsets $S_z\,(z = 1, \ldots, m)$: $\forall$ edge $(i, j) \in E, i, j \in V$ and $j \in FL_i$ the following holds $i \in S_A$ and $j \in S_B$ with $A \leq B$

(4.a)

$$\tilde{t}S_z \leq \tilde{t}_{sum} \quad \text{for all } z = 1, \ldots, m \tag{4.b}$$

where $\tilde{t}_{sum} = \sum_{j=1}^{n} \tilde{t}_j$ is the sum of all the tasks' fuzzy processing times and FL_i is the set of immediate followers (successors) of task i.

2. MO SALBP-2 version 2:

$$\text{Minimize } F_1 = w_1 \cdot \tilde{c} + w_2 \cdot \widetilde{BD} \tag{5}$$

subject to the constraints (4.a) and (4.b)

Constraints (4.a) and (4.b) ensure the feasibility of an ALB solution. In particular, constraint (4.a) guarantees the feasible assignment of the tasks to the m stations. That is, each task is assigned to exactly one station, and the successors of any task i are not assigned to an earlier station than that of i. Note that, (i, j) denotes an edge between i and j, with j being the immediate successor of i. Constraint (4.b) ensure that the station times of all the stations do not exceed the line's total processing time (t_{sum}). The weights w_1 and w_2 in Eqs. (4) and (5), specify the relative importance of the corresponding objectives. The determination of the suitable values for these weights is in general a difficult task and constitutes a critical research question in MO optimization. This issue will be discussed deeper in sect."The proposed solution model for the MO fuzzy SALBP-2" of this study.

Arithmetics and ranking fuzzy numbers

The purpose of fuzzy data approach is to represent more realistic situations, where data are imprecise, uncertain or almost unavailable (Kaufmann and Gupta 1985). The membership function $\mu_{\tilde{A}}(x)$ of a fuzzy data $\tilde{A}$ represents the grade of satisfaction of a decision maker for the completion time of that scheduling. In this work, the fuzziness of data is represented by Triangular Fuzzy Numbers (TFNs), as shown in Fig. 1. A TFN $\tilde{A}$ is denoted as a triplet $(\alpha_1, \alpha_2, \alpha_3)$.

Arithmetics of TFNs are performed as following:

$$\begin{aligned}
\tilde{A} + \tilde{B} &= (\alpha_1 + \beta_1, \alpha_2 + \beta_2, \alpha_3 + \beta_3) \\
\tilde{A} - \tilde{B} &= (\alpha_1 - \beta_3, \alpha_2 - \beta_2, \alpha_3 - \beta_1) \\
\tilde{A} \times \tilde{B} &= (\alpha_1 \cdot \beta_1, \alpha_2 \cdot \beta_2, \alpha_3 \cdot \beta_3) \\
\tilde{A}/\tilde{B} &= (\alpha_1/\beta_3, \alpha_2/\beta_2, \alpha_3/\beta_1)
\end{aligned} \tag{6}$$

To compare the fuzzy numbers, some criteria of ranking fuzzy sets are presented:

- The greatest associate ordinary number:

$$F_1\left(\tilde{A}\right) = \frac{\alpha_1 + 2\alpha_2 + \alpha_3}{4} \tag{7}$$

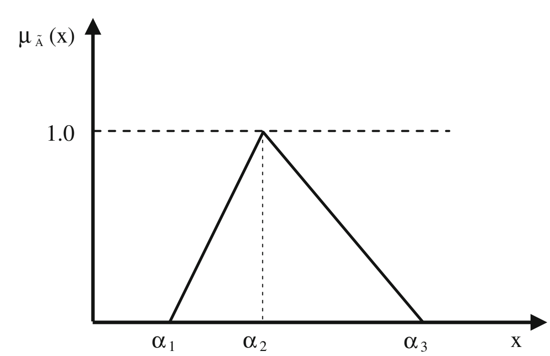

Fig. 1 A typical Triangular Fuzzy Number

- The best maximum presumption (the mode):

$$F_2\left(\tilde{A}\right) = \alpha_2 \tag{8}$$

- The divergence (the distance between two end-points):

$$F_3\left(\tilde{A}\right) = \alpha_3 - \alpha_1 \tag{9}$$

Consider a set Q composed of TFNs $\tilde{A}_i, i = 1, 2, \ldots, n$. We define $\tilde{A}^*$ as a major TFN that dominates all the others in some criterion, in Q, that is, $\tilde{A}^* = \max Q$ (the operator max is the discrete maximum). The decision maker chooses some criteria and determines its order of dominance. If the first criterion can not determine the major TFN, go to second criterion, and so on. On the contrary, we call a minor TFN if TFN is dominated by all others in Q and this operation is represented as min.

The proposed solution model for the MO fuzzy SALBP-2

GAs (Holland 1975; Goldberg 1989) are probabilistic search methods that employ search techniques inspired by Darwin's evolutionary theory based on the principles and mechanisms of natural selection and the survival of the fittest. GAs employ a random yet directed, search for finding the globally optimal solution. They have the advantage over the gradient descent techniques that they do not require the derivative of the objective function and the search is not biased towards the locally optimal solution. In contrast to random sampling algorithms, GAs have the ability to direct the search towards relatively promising regions in the problem's search space. In addition, they have been empirically proven very effective in solving a large number of complex combinatorial optimization problems.

The architecture of any GA consists of the following five basic components:

a) A representation mechanism, i.e., a way of encoding the phenotypes to genotypes.
b) A decoding mechanism, i.e., a way of mapping the phenotypes to actual solutions of the optimization problem under consideration.
c) An evaluation mechanism, i.e., a way of computing the cost-function for each genotype.
d) A way to generate the initial population of the genotypes.
e) Generate new genotypes by applying variation operators on the entire population.

The representation mechanism

In this work a real-valued GA was adopted for use, i.e., genotypes are represented by floating-point vectors. Therefore, since actual ALBP solutions are represented by strings of integers (Scholl 1999), an appropriate mapping is needed from the genotypic state-level (the real-valued vectors) to the phenotypic level (the actual ALB solutions). To achieve this mapping a simple yet effective topological ordering scheme has been developed based on the relative priorities impose by the components of a genotype. Assuming a n-task ALBP with precedence relations given by a DAG $G = (V,E)$, the developed encoding scheme consists of generating a topological sort of G from a specific n-dimensional floating-point vector ψ (genotype). Each vector's component ψ_i $(i = 1,n)$ represents the relative priority of task $i\,(i \in V)$. The topological sort is therefore a ranking of all the tasks according to their priorities in an appropriate order to meet the precedence constraints. This mechanism is implemented using the following procedure:

Procedure Topological_ordering_encoding
begin
  Set $V' = \varnothing$ // with $V' \subseteq V$//
  repeat
    for all $j \in V$ **do**
        if j has no predecessors **then** $V' = V \cup \{j\}$,
          i.e., insert j into the set V'.
    Determine the gene ψ_i of ψ with the maximum value
          for all $i \in V'$
    Insert task i into the next available position in the
        partial schedule (PS).
    $V' = V'\ \{i\}$, i.e., remove task i from V'.
  until PS has been completed
  return *PS*
end

In each step, the tasks with no predecessors are identified and put in set V'. Then, the task in V' having the highest gene's value in ψ is selected, removed from V', and placed in the next available position of *PS*. The process is repeated until the completion of *PS*.

Let us see how this topological ordering works on genotype $\psi = (0.32, 0.83, 0.05, 0.24, 0.17, 0.45, 0.09, 0.61)$ concerning the 8-task ALBP shown in Fig. 2. The first position of array *PS* is taken by task 1 (i.e., PS[1] = 1) since this is the only task with no predecessors. Task 1 is then cut from DAG and the next task with no predecessors is task 2, thus PS[2] = 2. Then, the two tasks 3 and 4 are candidate for the 3rd location of *PS*. The priorities for these tasks are 0.05 and 0.24, respectively, and therefore, *PS*[3] = 4 since task 4 has the highest priority, consequently PS[4] = 3. Finally, the ALB solution corresponding to ψ will be (1, 2, 4, 3, 6, 5, 7, 8). Figure 3 displays the detailed step-by-step process for

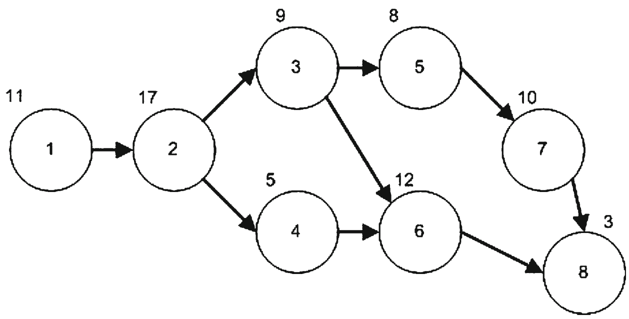

Fig. 2 A precedence graph for an 8-task ALBP

constructing the specific feasible ALB solution. One can see from this figure the partial topological sort, the cut (dark long dashed lines) and the eligible nodes, as well as, the contents of the partial schedule solution *PS*.

The decoding mechanism

Once a specific genotype is encoded into a phenotype, a way is needed to assign the tasks in the generated task-sequence into the stations. In this work, a scheme proposed by Kim et al. (1996) was adopted; this scheme was found to be superior to other traditional schemes (see Scholl 1999) in terms of quality of solutions. This scheme works as follows:

- **Step 1:** Set $\tilde{c}$ initially equal to the theoretical minimum fuzzy cycle time, i.e. $\tilde{c}_{th} = \tilde{t}_{sum}/m$.
- **Step 2:** Assign as many tasks as possible into the first *m-1* workstations. Assign all the remaining tasks to the last workstation, *m*.
- **Step 3:** Calculate the fuzzy work load $\widetilde{W}_z$ for each workstation z ($z = 1, 2, \ldots, m$), and the potential fuzzy workload $\widetilde{PW}_z$ ($z = 1, 2, \ldots, m-1$) as follows: $\widetilde{W}_z =$ the fuzzy station time $\widetilde{tS}_z$ ($z = 1, 2, \ldots, m$). $\widetilde{PW}_z = \widetilde{tS}_z +$ the processing time of the first task assigned to (z+1)st station ($z = 1, 2, \ldots, m-1$).
- **Step 4:** Set $\tilde{c}_w = \max\left\{\widetilde{W}_1, \widetilde{W}_2, \ldots, \widetilde{W}_m\right\}$ and $\tilde{c} = \min\left\{\widetilde{PW}_1, \widetilde{PW}_2, \ldots, \widetilde{PW}_{m-1}\right\}$
- **Step 5: if** $\tilde{c}_w \succ \tilde{c}$ **then goto** Step 2 **else** Return $\tilde{c}_w$

The evaluation mechanism

The evaluation mechanism concerns the computation of the objective function for each candidate solution. The objective functions to be minimized are described by Eqs. (4) and (5). The weights w_1 and w_2 specify the relative importance of the corresponding objectives. The determination of the suitable values for these weights is in general a difficult task and constitutes a critical research question in MO optimization problems.

In the literature, there are at least three general methods to compute the weights w_i ($i = 1, \ldots, k$) for a weighted-sum objective function with k objectives: the fixed-, the random- and the adaptive-weight method. The first one uses constant weights satisfying the relation:

$$\sum_{i=1}^{k} w_i = 1 \tag{10}$$

where $w_i > 0$ for all $i = 1, \ldots, k$.

However, it has been shown (Murata et al. 1996) that the search direction is fixed when using constant weights within an evolutionary algorithm; thus, it is difficult for the search process to obtain a variety of non-dominated solutions. To alleviate this drawback, the use of random weights was proposed according to the following formula:

$$w_i = \frac{\text{random}_i}{\text{random}_1 + \text{random}_2 +, \ldots, +\text{random}_k} \tag{11}$$

where random_i ($i = 1, \ldots, k$) are non-negative random numbers.

Alternatively, an adaptive weight approach has been proposed by Gen and Cheng (2000) that readjusts the weights by utilizing some useful information from the current population. The weights are given by the formula:

$$w_i = \frac{1}{z_i^{\max} - z_i^{\min}} \tag{12}$$

where $z_i^{\max}$ and $z_i^{\min}$, for $i = 1, \ldots, k$ are the maximum and the minimum values for the ith objective in the population.

The initial population

Usually, the initial population is randomly created in order to uniformly distribute the selected chromosomes (solutions over the search space). In some cases, solutions obtained from another optimization algorithm are used to seed the initial population (Goldberg 1989). Although this bears the risk of misguiding the optimization process toward local optima, it has been empirically proven that for some problems, the use of a seeding mechanism results to a much powerful optimizer. Our approach is motivated by the idea of case retrieval (Oman and Cunningham 2001); that is, by the expectation that seeding the initial population with near-optimum solutions will speed up the GA by starting the search in promising regions of the search space. The most common way to achieve this mechanism is to apply a problem-specific heuristic and generate a fairly good solution to the problem under consideration. This solution is copied into the initial population which then undergoes the conventional stages of a canonical GA.

Re-initializing the entire population during the genetic search is another common challenge in designing competent GAs (Goldberg 1989; Michalewitz 1996). This mech-

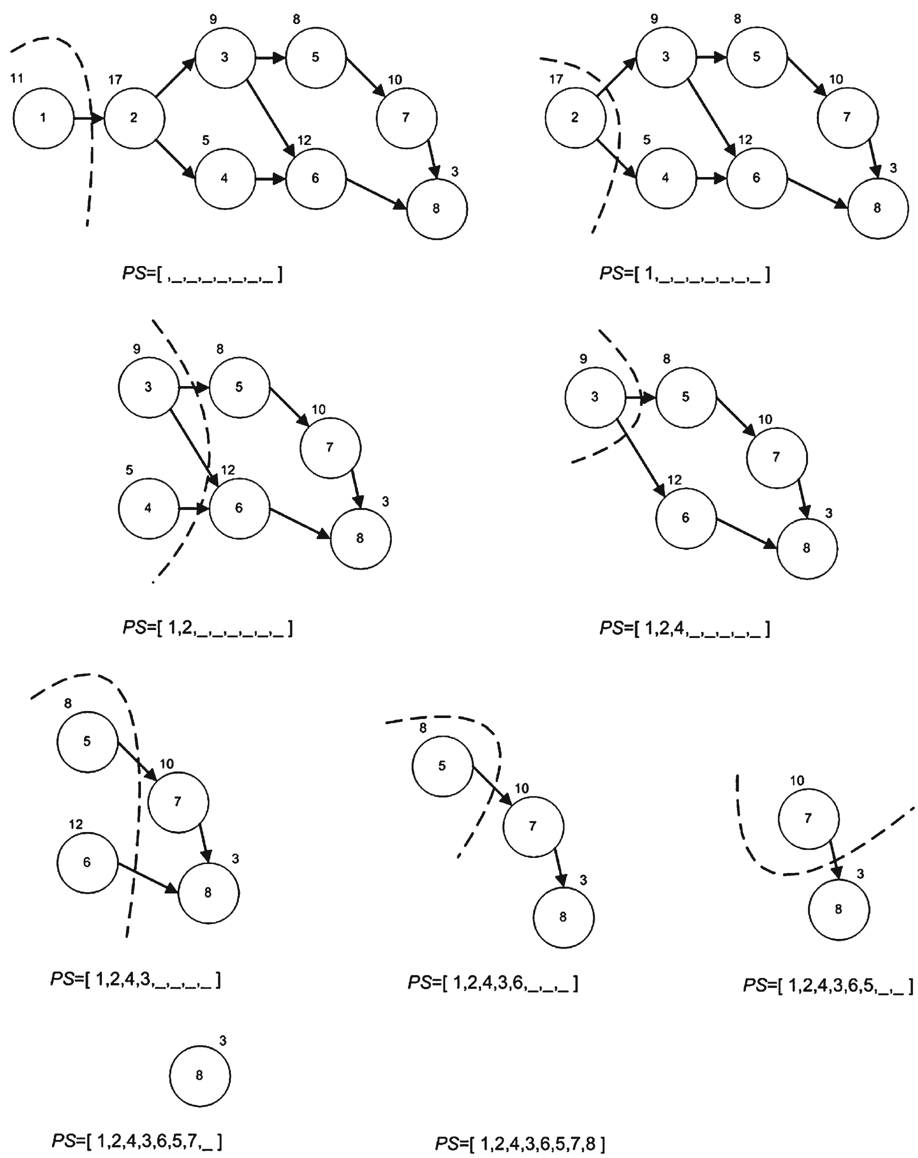

Fig. 3 The application of topological ordering encoding method on genotype $\psi =$ (0.32, 0.83, 0.05, 0.24, 0.17, 0.45, 0.09, 0.61)

anism is performed with the hope to improve the diversity of the population and avoid premature convergence to local optima solutions. Following the same line of thought, the developed approach uses a hybrid GA that combines *seeding* and *re-initialization* during the genetic search. The hybrid GA is performed in two successive runs. In the first run, the search starts from a randomly generated population of solutions. This population is then evolved over successive iterations undergoing selection and random variation operators (see sub-sections "The variation and selection operators" below). In every iteration, the algorithm keeps truck for the best-so-far (***bsf***) solution. In the second run, the entire population is re-initialized by new, randomly created individual solutions, and seeded by the ***bsf*** solution determined in the previous run. Figure 4 presents schematically the mechanics of the proposed approach. Therefore, instead of using a simple problem-specific heuristic, e.g., a common ALB priority rule (Scholl and Voß 1996) to seed the GA's initial population (as it was explained above this is the traditional way of hybridizing a GA); the proposed approach applies a GA to seed the initial population of a second GA (the two GAs are identical). Multiple preliminary experiments showed that this new hybridization scheme results to a much more powerful optimizer for the examined ALBP.

The variation and selection operators

In each generation, a subset of the entire population is replaced using a suitable parent selection strategy. *Selection* allocates more copies of highly fit individuals (solu-

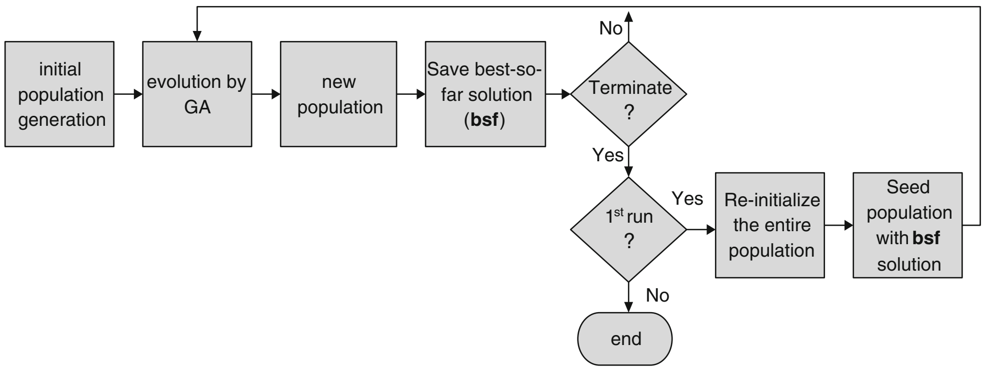

Fig. 4 The mechanics of the proposed evolutionary approach

tions with higher fitness values), in an analogy to the *survival-of-the-fittest* mechanism. Offspring are then created by applying variation operators namely, *crossover* and *mutation* on the selected population subset. The role of *crossover* is to join together parts of two or more parental solutions in order to produce new, possibly better solutions (offspring). The offspring will differ from their parents but they will instead combine parental features (genetic material) in a novel manner. *Mutation* is applied in order to inject new genetic material into the population and thereby avoid premature convergence to local minima. In the proposed approach *selection* was accomplished using the well known roulette wheel procedure (Goldberg 1989). *Crossover* and *mutation* were accomplished by one-point crossover, and random mutation, respectively (Michalewitz 1996).

Numerical results and discussion

The performance of f-MOGA was evaluated over known ALBP benchmarks taken from the open literature (Scholl 1999). In the experiments, we include the available test instances concerning the following three ALBPs: **Sawyer** ($n=30$, in#=8), **Kilbridge** ($n=45$, *in#*=9) and **Tonge** ($n=70$, *in#*=23). n denotes the number of the tasks included in the corresponding precedence graphs, and *in#* the number of the test instances included in the specific ALBP. The simulations were implemented in Matlab and run on a Pentium IV 2.13 GHz core2 PC. Considering fuzziness for the processing times, fuzzy data are represented by triangular fuzzy membership functions. 10 runs (starting each time from a different random number seed) on each problem instance were performed. The best solutions obtained after these runs was retained and presented in Tables 1–6 below. Furthermore, 3 different versions of f-MOGA were evaluated differ in the way weights (w_1 and w_2) in the weighted-sum fitness function are estimated [see Eqs. (10)–(12)]. For the case of the fixed-weight method we set $w_1 = w_2 = 0.5$.

After extended experimentation, the following settings for f-MOGA's control parameters were adopted: population size = 50, maximum number of generations = 1500, a random crossover rate taken values in the range [0.6, 0.9], and a random mutation rate taken values in the range [0.06, 0.1]. That is, crossover and mutation rates are not fixed during the algorithm's evolution; instead, they change stochastically within the above permitted bounds. As it was empirically verified, this novelty gives a significant impetus to f-MOGA by speeding its rate of convergence.

Table 1 shows the experimental results for MO SALBP-2 version 1 over the Sawyer ALBP test instances. Note that column $\boldsymbol{m}$ denotes the number of workstations associated with each one of the 8 instances included in Sawyer problem. Hence, for the first instance the work load ($n=30$) must be assigned to 7 workstations, for the second instance the work load must be assigned to 8 workstations, and so on. Column **BE** denotes the balance efficiency (Eq. (1)) of the line associated to the generated best solutions. Tables 2 and 3, display the experimental results for MO SALBP-2 version 1, over Kilbridge and Tonge ALBPs, respectively.

The experimental results concerning MO SALBP-2 version 2 are reported in Tables 4–6. In particular, Table 4 displays the results concerning Sawyer ALBPs, while Tables 5 and 6 the results concerning Kilbridge and Tonge problems, respectively. Note that, the secondary objective to be minimized is now $\widetilde{\boldsymbol{SX}}$ (given Eq. (3)).

Comparing the results yielded by the three different versions of f-MOGA, one can observe that f-MOGA with the fixed-weight method outperforms the two other methods (with the regard to the solutions quality obtained) for both versions of the examined MO SALBP-2. This is established from the fact that, in most of the experiments f-MOGA+ fixed-weight method attained to obtain solutions with fuzzy balance efficiency near to 1.0 (meaning almost a perfect balance of the line). As one can observe from the above tables, the associated results obtained by the other two versions of f-MOGA were of inferior quality (with regard to $\widetilde{\boldsymbol{BE}}$ measure).

Table 1 Minimizing cycle time and smoothness index for the Sawyer ALBP test instances

Fitness weighted-method

m	Fixed			Random			Adaptive		
	$\tilde{c}$	$\widetilde{SX}$	$\widetilde{BE}$	$\tilde{c}$	$\widetilde{SX}$	$\widetilde{BE}$	$\tilde{c}$	$\widetilde{SX}$	$\widetilde{BE}$
7	(44.50, 48.00, 51.40)	(0.00, 5.10, 23.56)	(0.83, 0.96, 1.00)	(45.90, 50.00, 54.70)	(0.00, 11.92, 32.92)	(0.78, 0.93, 1.00)	(24.00, 26.00, 28.00)	(3.74, 12.57, 26.86)	(0.75, 0.89, 1.00)
8	(38.70, 41.00, 44.20)	(0.00, 2.00, 19.78)	(0.84, 0.99, 1.00)	(42.00, 44.00, 48.30)	(4.10, 13.27, 32.90)	(0.77, 0.92, 1.00)	(38.70, 41.00, 44.20)	(0.00, 3.16, 19.88)	(0.84, 0.99, 1.00)
9	(34.20, 37.00, 40.00)	(0.00, 4.36, 20.83)	(0.83, 0.97, 1.00)	(37.00, 41.00, 45.00)	(5.28, 18.74, 36.94)	(0.74, 0.88, 1.00)	(34.20, 37.00, 40.00)	(0.00, 4.58, 20.85)	(0.83, 0.97, 1.00)
10	(32.10, 34.00, 36.00)	(0.00, 5.83, 21.46)	(0.82, 0.95, 1.00)	(37.50, 41.00, 44.60)	(29.61, 38.94, 53.44)	(0.67, 0.79, 0.94)	(30.80, 34.00, 38.40)	(0.00, 7.35, 27.47)	(0.78, 0.95, 1.00)
11	(29.60, 32.00, 33.50)	(0.00, 9.16, 21.72)	(0.81, 0.92, 1.00)	(32.10, 34.00,36.50)	(10.07, 18.81, 33.03)	(0.74, 0.87, 1.00)	(29.60, 32.00, 33.50)	(2.33, 10.49, 21.91)	(0.81, 0.92, 1.00)
12	(26.90, 29.00, 31.10)	(0.60, 8.49, 22.20)	(0.80, 0.93, 1.00)	(28.80, 32.00, 34.90)	(12.03, 22.98, 37.64)	(0.71, 0.84, 1.00)	(27.20, 30.00, 33.40)	(1.92, 12.41, 30.31)	(0.74, 0.90, 1.00)
13	(24.40, 27.00, 29.20)	(0.00, 8.78, 22.97)	(0.78, 0.92, 1.00)	(28.20, 30.00, 33.70)	(18.62, 26.23, 42.85)	(0.68, 0.83, 0.96)	(25.10, 27.00, 29.40)	(2.81, 9.75, 23.99)	(0.78, 0.92, 1.00)
14	(23.60, 25.00, 28.10)	(1.30, 8.60, 26.10)	(0.76, 0.93, 1.00)	(24.40, 27.00, 29.30)	(7.79, 18.28, 31.69)	(0.73, 0.86, 1.00)	(24.20, 26.00, 27.60)	(6.87, 14.42, 25.29)	(0.77, 0.89, 1.00)

Table 2 Minimum cycle time and smoothness index for Kilbridge ALBP test instances

Fitness weighted-method

m	Fixed			Random			Adaptive		
	$\tilde{c}$	$\widetilde{SX}$	$\widetilde{BE}$	$\tilde{c}$	$\widetilde{SX}$	$\widetilde{BE}$	$\tilde{c}$	$\widetilde{SX}$	$\widetilde{BE}$
3	(173.50, 184.00, 194.50)	(0.00, 0.00, 39.85)	(0.88, 1.00, 1.00)	(173.50, 186.00, 197.40)	(0.00, 4.47, 45.01)	(0.87, 0.99, 1.00)	(173.50, 185.00, 194.50)	(0.00, 2.24, 39.85)	(0.88, 0.99, 1.00)
4	(130.00, 138.00, 146.10)	(0.00, 0.00, 34.94)	(0.88, 1.00, 1.00	(134.40, 144.00, 154.00)	(0.00, 16.67, 51.94)	(0.84, 0.96, 1.00)	(132.30, 141.00, 147.90)	(0.00, 7.07, 38.79)	(0.87, 0.98, 1.00)
5	(104.70, 111.00, 116.30)	(0.00, 1.73, 30.02)	(0.89, 0.99, 1.00)	(107.3, 115.00, 122.20)	(0.00, 13.89, 43.93)	(0.84, 0.96, 1.00)	(103.30, 111.00, 119.30)	(0.00, 2.24, 36.7)	(0.86, 0.99, 1.00)
6	(85.60, 92.00, 98.50)	(0.00, 0.00, 31.21)	(0.87, 1.00, 1.00)	(97.40, 104.00, 110.90)	(31.05, 46.11, 70.32)	(0.77, 0.88, 1.00)	(87.60, 93.00, 97.20)	(0.00, 3.46, 28.09)	(0.88, 0.99, 1.00)
7	(74.50, 80.00, 83.70)	(0.00, 3.46, 27.05)	(0.88, 0.98, 1.00)	(82.40, 88.00, 93.30)	(15.68, 31.94, 56.24)	(0.79, 0.90, 1.00)	(73.60, 80.00, 85.50)	(0.00, 4.00, 31.78)	(0.86, 0.99, 1.00)
8	(67.00, 70.00, 73.10)	(0.00, 3.46, 25.10)	(0.88, 0.99, 1.00)	(70.90, 76.00, 81.80)	(9.14, 24.94, 51.43)	(0.79, 0.91, 1.00)	(65.80, 71.00, 76.60)	(0.00, 8.94, 35.33)	(0.84, 0.97, 1.00)
9	(60.40, 63.00, 66.00)	(0.00, 6.56, 26.89)	(0.87, 0.97, 1.00)	(60.90, 65.00, 69.30)	(0.80, 13.75, 37.22)	(0.83, 0.94, 1.00)	(58.50, 63.00, 67.90)	(0.00, 7.42, 32.61)	(0.84, 0.97, 1.00)
10	(52.30, 56.00, 60.10)	(0.00, 3.46, 27.4)	(0.86, 0.99, 1.00)	(57.40, 62.00, 66.20)	(14.07, 28.53, 49.64)	(0.78, 0.89, 1.00)	(53.40, 58.00, 61.50)	(0.00, 10.2, 32.05)	(0.84, 0.95, 1.00)
11	(52.00, 55.00, 57.10)	(0.10, 17.00, 34.76)	(0.82, 0.91, 1.00)	(56.40, 60.00, 63.50)	(35.50, 46.95, 64.05)	(0.74, 0.84, 0.95)	(52.00, 55.00, 57.10)	(7.19, 20.86, 36.73)	(0.82, 0.91, 1.00)

Table 3 Minimum cycle time and smoothness index for Tonge ALBP test instances

Fitness weighted-method

m	Fixed			Random			Adaptive		
	$\tilde{c}$	$\widetilde{SX}$	$\widetilde{BE}$	$\tilde{c}$	$\widetilde{SX}$	$\widetilde{BE}$	$\tilde{c}$	$\widetilde{SX}$	$\widetilde{BE}$
3	(1147.70, 1171.00, 1200.40)	(0.00, 2.24, 103.51)	(0.95, 1.00, 1.00)	(1142.80, 1178.00, 1220.90)	(0.00, 17.20, 138.89)	(0.93, 0.99, 1.00)	(1144.40, 1173.00, 1207.40)	(0.00, 7.28, 115.90)	(0.94, 0.99, 1.00)
4	(859.20, 879.00, 905.00)	(0.00, 3.74, 98.89)	(0.95, 1.00, 1.00)	(880.80, 899.00, 921.40)	(14.4, 61.50, 138.93)	(0.93, 0.98, 1.00)	(862.00, 882.00, 905.80)	(0.00, 12.25, 100.64)	(0.94, 0.99, 1.00)
5	(692.00, 704.00, 718.60)	(0.00, 5.48, 76.67)	(0.95, 1.00, 1.00)	(709.80, 727.00, 749.50)	(10.93, 67, 150.06)	(0.91, 0.97, 1.00)	(682.80, 705.00, 732.80)	(0.00, 9.11, 108.21)	(0.93, 0.99, 1.00)
6	(579.80, 590.00, 601.60)	(0.00, 16.31, 77.22)	(0.95, 0.99, 1.00)	(587.90, 606.00, 630.00)	(47.40, 83.83, 158.86)	(0.91, 0.97, 1.00)	(579.80, 590.00, 601.60)	(0.00, 19.03, 77.42)	(0.95, 0.99, 1.00)
7	(494.30, 504.00, 517.00)	(0.00, 10.68, 75.08)	(0.95, 0.99, 1.00)	(520.20, 533.00, 547.70)	(101.05, 137.70, 190.79)	(0.89, 0.94, 0.99)	(496.80, 508.00, 521.20)	(0.00, 22.23, 86.74)	(0.94, 0.99, 1.00)
8	(432.80, 443.00, 455.90)	(0.00, 14.97, 80.06)	(0.94, 0.99, 1.00)	(453.50, 463.00, 475.30)	(74.80, 104.04, 153.91)	(0.90, 0.95, 1.00)	(436.00, 448.00, 461.70)	(7.30, 38.39, 99.07)	(0.93, 0.98, 1.00)
9	(385.90, 394.00, 403.80)	(0.00, 14.63, 71.01)	(0.94, 0.99, 1.00)	(427.40, 435.00, 446.10)	(153.51, 192.36, 241.49)	(0.85, 0.90, 0.94)	(386.80, 395.00, 404.80)	(0.00, 20.07, 74.82)	(0.94, 0.99, 1.00)
10	(345.60, 355.00, 366.40)	(0.00, 18.33, 77.44)	(0.93, 0.99, 1.00)	(363.30, 370.00, 380.50)	(32.86, 71.72, 127.03)	(0.90, 0.95, 1.00)	(348.30, 362.00, 377.80)	(0.70, 42.47, 114.19)	(0.91, 0.97, 1.00)
11	(319.20, 325.00, 333.90)	(0.00, 24.96, 77.88)	(0.93, 0.98, 1.00)	(331.30, 344.00, 357.30)	(56.80, 104.83, 165.32)	(0.87, 0.93, 0.99)	(323.40, 329.00, 336.90)	(11.28, 43.12, 90.58)	(0.92, 0.97, 1.00)
12	(292.60, 297.00, 304.30)	(3.90, 19.60, 67.37)	(0.94, 0.98, 1.00)	(314.50, 321.00, 330.30)	(101.39, 132.30, 176.78)	(0.86, 0.91, 0.96)	(298.60, 308.00, 320.00)	(40.75, 74.16, 130.07)	(0.89, 0.95, 1.00)
13	(271.20, 275.00, 281.40)	(6.72, 26.29, 68.17)	(0.94, 0.98, 1.00)	(314.20, 323.00, 332.80)	(299.95, 324.75, 357.83)	(0.79, 0.84, 0.89)	(273.40, 279.00, 286.20)	(27.19, 47.82, 88.91)	(0.92, 0.97, 1.00)
14	(253.20, 257.00, 262.40)	(4.32, 28.11, 69.24)	(0.93, 0.98, 1.00)	(280.40, 286.00, 291.90)	(119.2, 158.11, 197.22)	(0.84, 0.88, 0.92)	(258.60, 264.00, 272.60)	(29.64, 62.48, 112.03)	(0.90, 0.95, 1.00)
15	(234.10, 241.00, 249.10)	(3.40, 31.64, 82.68)	(0.92, 0.97, 1.00)	(279.30, 283.00, 288.10)	(223.09, 252.11, 284.71)	(0.79, 0.83, 0.86)	(250.30, 255.00, 263.30)	(76.51, 104.71, 150.75)	(0.87, 0.92, 0.96)
16	(220.30, 226.00, 233.70)	(6.57, 34.79, 82.77)	(0.92, 0.97, 1.00)	(238.60, 245.00, 253.40)	(110.45, 141.59, 183.99)	(0.84, 0.90, 0.95)	(230.00, 234.00, 240.00)	(82.16, 98.09, 128.64)	(0.89, 0.94, 0.98)
17	(213.30, 218.00, 225.50)	(20.59, 54.17, 103.30)	(0.89, 0.95, 1.00)	(235.30, 240.00, 245.70)	(160.92, 189.88, 223.97)	(0.82, 0.86, 0.90)	(219.50, 223.00, 228.90)	(80.29, 101.69, 135.80)	(0.88, 0.93, 0.97)
18	(203.90, 207.00, 210.70)	(27.30, 56.48, 90.48)	(0.90, 0.94, 0.99)	(221.10, 228.00, 237.10)	(186.29, 214.15, 253.95)	(0.80, 0.86, 0.91)	(204.40, 211.00, 219.20)	(86.98, 111.00, 149.47)	(0.87, 0.92, 0.98)
19	(189.70, 194.00, 199.50)	(28.35, 53.81, 91.11)	(0.90, 0.95, 1.00)	(210.20, 216.00, 223.00)	(185.00, 210.85, 244.16)	(0.81, 0.86, 0.91)	(200.50, 207.00, 214.60)	(102.95, 133.33, 173.95)	(0.84, 0.89, 0.95)
20	(183.70, 189.00, 196.20)	(59.60, 84.56, 125.26)	(0.87, 0.93, 0.98)	(209.70, 215.00, 221.90)	(222.18, 248.86, 283.72)	(0.77, 0.82, 0.86)	(185.00, 190.00, 196.80)	(68.15, 91.20, 129.74)	(0.87, 0.92, 0.98)
21	(175.50, 180.00, 186.70)	(39.39, 68.56, 113.76)	(0.87, 0.93, 0.98)	(202.30, 210.00, 219.70)	(245.39, 278.41, 323.32)	(0.74, 0.80, 0.85)	(182.30, 185.00, 188.10)	(93.70, 115.38, 140.3)	(0.87, 0.90, 0.94)
22	(167.30, 172.00, 176.80)	(32.91, 65.15, 102.65)	(0.88, 0.93, 0.98)	(209.00, 217.00, 226.20)	(327.58, 364.82, 409.30)	(0.69, 0.74, 0.79)	(172.40, 177.00, 182.70)	(83.19, 109.73, 144.89)	(0.85, 0.90, 0.95)
23	(159.60, 162.00, 165.30)	(32.51, 55.37, 85.97)	(0.90, 0.94, 0.99)	(186.40, 192.00, 198.70)	(201.01, 236.39, 277.26)	(0.75, 0.79, 0.84)	(161.30, 166.00, 171.90)	(59.72, 87.49 124.75)	(0.87, 0.92, 0.98)
24	(154.00, 156.00, 157.80)	(44.23, 62.98, 84.14)	(0.90, 0.94, 0.98)	(175.80, 182.00, 190.10)	(182.10, 218.77, 265.63)	(0.75, 0.80, 0.86)	(157.20, 159.00, 163.00)	(93.00, 104.86, 129.02)	(0.87, 0.92, 0.96)
25	(154.00, 156.00, 157.80)	(62.01, 88.76, 113.31)	(0.87, 0.90, 0.94)	(172.80, 178.00, 185.10)	(221.47, 251.04, 290.60)	(0.74, 0.79, 0.84)	(155.20, 158.00, 162.80)	(96.62, 118.59, 151.06)	(0.84, 0.89, 0.93)

Table 4 Minimum cycle time and balance delay for Sawyer's ALBP test instances

Fitness weighted-method									
m	Fixed			Random			Adaptive		
	$\tilde{c}$	$\widetilde{BD}$	$\widetilde{BE}$	$\tilde{c}$	$\widetilde{BD}$	$\widetilde{BE}$	$\tilde{c}$	$\widetilde{BD}$	$\widetilde{BE}$
7	(45.90, 48.00, 50.60)	(0.00, 12.00, 56.30)	(0.84, 0.96, 1.00)	(45.90, 50.00, 54.70)	(0.00, 26.00, 85.00)	(0.78, 0.93, 1.00)	(44.50, 48.00, 51.40)	(0.00, 12.00, 61.90)	(0.83, 0.96, 1.00)
8	(38.70, 41.00, 44.20)	(0.00, 4.00, 55.70)	(0.84, 0.98, 1.00)	(43.20, 47.00, 50.20)	(23.40, 52.00, 103.70)	(0.74, 0.86, 1.00)	(39.80, 42.00, 45.90)	(0.00, 12.00, 69.30)	(0.81, 0.96, 1.00)
9	(34.70, 37.00, 39.80)	(0.00, 9.00, 60.30)	(0.83, 0.97, 1.00)	(36.70, 39.00, 41.90)	(7.40, 27.00, 79.20)	(0.79, 0.92, 1.00)	(33.90, 37.00, 40.80)	(0.00, 9.00, 69.30)	(0.81, 0.97, 1.00)
10	(31.50, 34.00, 36.00)	(0.00, 16.00, 62.10)	(0.83, 0.95, 1.00)	(36.10, 40.00, 43.80)	(38.00, 76.00, 140.10)	(0.68, 0.81, 0.98)	(31.30, 34.00, 37.50)	(0.00, 16.00, 77.10)	(0.79, 0.95, 1.00)
11	(29.60, 32.00, 33.50)	(0.00, 28.00, 70.60)	(0.81, 0.92, 1.00)	(32.30, 35.00, 38.10)	(24.90, 61.00, 121.20)	(0.71, 0.84, 0.99)	(29.60, 32.00, 33.50)	(3.20, 28.00, 70.60)	(0.81, 0.92, 1.00)
12	(26.70, 28.00, 30.10)	(0.40, 12.00, 63.30)	(0.82, 0.96, 1.00)	(28.80, 32.00, 34.90)	(23.60, 60.00, 120.90)	(0.71, 0.84, 1.00)	(28.20, 30.00, 33.70)	(6.10, 36.00, 106.50)	(0.74, 0.90, 1.00)
13	(24.30, 26.00, 28.40)	(0.00, 14.00, 71.30)	(0.81, 0.96, 1.00)	(28.90, 30.00, 32.50)	(42.00, 66.00, 124.60)	(0.71, 0.83, 0.94)	(24.90, 27.00, 28.80)	(2.60, 27.00, 76.50)	(0.79, 0.92, 1.00)
14	(22.50, 25.00, 27.50)	(4.80, 26.00, 87.10)	(0.77, 0.93, 1.00)	(26.40, 29.00, 31.60)	(36.20, 82.00, 144.50)	(0.67, 0.80, 0.95)	(23.60, 25.00, 28.10)	(5.50, 26.00, 95.50)	(0.70, 0.90, 1.00)

Table 5 Minimum cycle time and balance delay for Kilbridge's test instances

Fitness weighted-method									
m	Fixed			Random			Adaptive		
	$\tilde{c}$	$\widetilde{BD}$	$\widetilde{BE}$	$\tilde{c}$	$\widetilde{BD}$	$\widetilde{BE}$	$\tilde{c}$	$\widetilde{BD}$	$\widetilde{BE}$
3	(173.10, 184.00, 194.70)	(0.00, 0.00, 69.50)	(0.88, 1.00, 1.00)	(173.50, 186.00, 197.40)	(0.00, 6.00, 77.60)	(0.87, 0.99, 1.00)	(173.00, 185.00, 194.50)	(0.00, 3.00, 68.90)	(0.88, 0.99, 1.00)
4	(130.30, 138.00, 146.30)	(0.00, 0.00, 70.60)	(0.88, 1.00, 1.00)	(132.80, 143.00, 152.00)	(0.00, 20.00, 93.40)	(0.85, 0.97, 1.00)	(130.80, 139.00, 148.20)	(0.00, 4.00, 78.20)	(0.87, 0.99, 1.00)
5	(105.30, 111.00, 116.60)	(0.00, 3.00, 68.40)	(0.88, 0.99, 1.00)	(107.00, 115.00, 121.10)	(0.00, 23.00, 90.90)	(0.85, 0.96, 1.00)	(103.40, 112.00, 119.70)	(0.00, 8.00, 83.90)	(0.86, 0.99, 1.00)
6	(85.60, 92.00, 98.50)	(0.00, 0.00, 76.40)	(0.87, 1.00, 1.00)	(87.50, 94.00, 101.30)	(0.00, 12.00, 93.20)	(0.85, 0.98, 1.00)	(87.20, 93.00, 97.70)	(0.00, 6.00, 71.60)	(0.88, 0.99, 1.00)
7	(75.80, 80.00, 83.30)	(0.00, 8.00, 68.50)	(0.88, 0.99, 1.00)	(78.30, 84.00, 89.50)	(3.30, 36.00, 111.90)	(0.82, 0.94, 1.00)	(74.90, 81.00, 86.50)	(0.00, 15.00, 90.90)	(0.85, 0.97, 1.00)
8	(66.60, 70.00, 73.70)	(0.00, 8.00, 75.00)	(0.87, 0.99, 1.00)	(75.30, 80.00, 86.50)	(69.80, 88.00, 177.40)	(0.74, 0.86, 0.98)	(65.70, 70.00, 74.80)	(0.00, 8.00, 83.80)	(0.86, 0.99, 1.00)
9	(59.90, 63.00, 66.10)	(0.00, 15.00, 80.30)	(0.87, 0.97, 1.00)	(66.60, 71.00, 75.60)	(43.60, 87.00, 165.80)	(0.76, 0.86, 0.98)	(60.10, 64.00, 67.60)	(0.00, 24.00, 93.80)	(0.85, 0.96, 1.00)
10	(52.30, 56.00, 60.10)	(0.00, 8.00, 86.40)	(0.86, 0.99, 1.00)	(54.20, 58.00, 62.60)	(8.40, 28.00, 111.40)	(0.82, 0.95, 1.00)	(54.70, 59.00, 63.50)	(12.20, 38.00, 120.40)	(0.81, 0.94, 1.00)
11	(52.00, 55.00, 57.10)	(0.10, 53.00, 113.50)	(0.82, 0.91, 1.00)	(53.20, 58.00, 61.80)	(24.80, 86.00, 165.20)	(0.76, 0.87 1.00)	(52.00, 55.00, 57.10)	(13.30, 53.00, 113.50)	(0.82, 0.91, 1.00)

Table 6 Minimum cycle time and balance delay for Tonge's test instancesc

Fitness weighted-method									
m	Fixed			Random			Adaptive		
	$\tilde{c}$	$\widetilde{BD}$	$\widetilde{BE}$	$\tilde{c}$	$\widetilde{BD}$	$\widetilde{BE}$	$\tilde{c}$	$\widetilde{BD}$	$\widetilde{BE}$
3	(1147.70, 1171.00, 1200.40)	(0.00, 3.00, 178.60)	(0.95, 1.00, 1.00)	(1142.80, 1178.00, 1220.90)	(0.00, 24.00, 240.10)	(0.93, 0.99, 1.00)	(1147.70, 1171.00, 1200.40)	(0.00, 3.00, 178.60)	(0.95, 1.00, 1.00)
4	(859.20, 879.00, 905.00)	(0.00, 6.00, 197.40)	(0.95, 1.00, 1.00)	(879.40, 900.00, 925.10)	(19.90, 90.00, 277.80)	(0.92, 0.98, 1.00)	(856.90, 879.00, 907.10)	(0.00, 6.00, 205.80)	(0.94, 1.00, 1.00)
5	(692.00, 704.00, 718.60)	(0.00, 10.00, 170.40)	(0.95, 1.00, 1.00)	(700.50, 723.00, 754.40)	(4.10, 105.00, 349.40)	(0.91, 0.97, 1.00)	(682.90, 705.00, 730.70)	(0.00, 15.00, 230.90)	(0.94, 1.00, 1.00)
6	(579.80, 590.00, 601.60)	(0.00, 30.00, 187.00)	(0.95, 1.00, 1.00)	(589.20, 600.00, 612.20)	(5.60, 90.00, 250.60)	(0.93, 0.98, 1.00)	(580.20, 591.00, 603.20)	(0.00, 36.00, 196.60)	(0.95, 0.99, 1.00)
7	(493.80, 504.00, 516.40)	(0.00, 18.00, 192.20)	(0.95, 1.00, 1.00)	(508.00, 523.00, 539.20)	(31.60, 151.00, 351.80)	(0.91, 0.96, 1.00)	(492.40, 506.00, 525.00)	(0.00, 32.00, 252.40)	(0.93, 0.99, 1.00)
8	(432.80, 443.00, 455.90)	(0.00, 34.00, 224.60)	(0.94, 0.99, 1.00)	(452.10, 469.00, 487.30)	(60.40, 242.00, 475.80)	(0.88, 0.94, 1.00)	(435.20, 447.00, 460.60)	(1.30, 66.00, 262.20)	(0.93, 0.98, 1.00)
9	(387.00, 394.00, 401.40)	(0.00, 36.00, 190.00)	(0.95, 0.99, 1.00)	(411.30, 418.00, 424.70)	(149.40, 252.00, 399.70)	(0.90, 0.93, 0.98)	(384.50, 397.00, 411.00)	(0.00, 63.00, 276.40)	(0.93, 0.98, 1.00)
10	(345.60, 355.00, 366.40)	(0.00, 40.00, 241.40)	(0.93, 0.98, 1.00)	(367.40, 380.00, 394.60)	(135.30, 290.00, 523.40)	(0.87, 0.92, 0.98)	(357.10, 364.00, 373.40)	(21.80, 130.00, 311.40)	(0.92, 0.96, 1.00)
11	(319.20, 325.00, 333.90)	(2.00, 65.00, 250.30)	(0.93, 0.98, 1.00)	(334.40, 343.00, 353.90)	(135.10, 263.00, 470.30)	(0.88, 0.93, 0.98)	(324.20, 331.00, 340.00)	(27.30, 131.00, 317.40)	(0.92, 0.96, 1.00)
12	(292.60, 297.00, 304.30)	(3.90, 54.00, 229.00)	(0.94, 0.98, 1.00)	(332.10, 338.00, 346.90)	(430.60, 546.00, 740.20)	(0.82, 0.87, 0.91)	(295.50, 301.00, 308.60)	(24.60, 102.00, 280.60)	(0.92, 0.97, 1.00)
13	(271.20, 275.00, 281.40)	(8.80, 65.00, 235.60)	(0.94, 0.98, 1.00)	(292.90, 301.00, 309.80)	(254.50, 403.00, 604.80)	(0.85, 0.90, 0.95)	(279.90, 287.00, 295.60)	(96.00, 221.00, 420.20)	(0.89, 0.94, 0.99)
14	(254.20, 258.00, 262.90)	(10.80, 102.00, 258.00)	(0.93, 0.97, 1.00)	(276.00, 281.00, 285.90)	(298.90, 424.00, 580.00)	(0.85, 0.89, 0.94)	(265.00, 269.00, 272.50)	(146.60, 256.00, 392.40)	(0.90, 0.93, 0.98)
15	(234.10, 241.00, 249.10)	(3.40, 105.00, 313.90)	(0.92, 0.97, 1.00)	(265.00, 269.00, 272.50)	(384.80, 525.00, 664.90)	(0.84, 0.87, 0.91)	(242.50, 248.00, 255.40)	(88.30, 210.00, 408.40)	(0.89, 0.94, 0.99)
16	(220.30, 226.00, 233.70)	(11.60, 106.00, 316.60)	(0.92, 0.97, 1.00)	(251.20, 258.00, 266.70)	(444.7, 618.00, 844.6)	(0.80, 0.85, 0.90)	(231.20, 236.00, 244.10)	(142.90, 266.00, 483.00)	(0.88, 0.93, 0.98)
17	(211.90, 218.00, 227.60)	(79.60, 196.00, 446.6)	(0.88, 0.95, 1.00)	(242.20, 247.00, 254.80)	(521.8, 689.00, 909.00)	(0.79, 0.84, 0.88)	(217.20, 221.00, 226.30)	(147.00, 247.00, 424.50)	(0.89, 0.93, 0.98)
18	(203.90, 207.00, 210.70)	(74.90, 216.00, 370.00)	(0.90, 0.94, 0.99)	(214.00, 220.00, 228.30)	(287.4, 450.00, 686.8)	(0.83, 0.89, 0.94)	(204.20, 212.00, 220.10)	(132.50, 306.00, 539.20)	(0.86, 0.92, 0.98)
19	(190.70, 194.00, 198.30)	(69.40, 176.00, 345.10)	(0.91, 0.95, 1.00)	(212.00, 216.00, 220.40)	(436.00, 594.00, 765.00)	(0.82, 0.86, 0.90)	(201.90, 208.00, 215.00)	(272.50, 442.00, 662.40)	(0.84, 0.89, 0.94)
20	(183.70, 189.00, 196.20)	(132.00, 270.00, 501.40)	(0.87, 0.93, 0.98)	(201.50, 208.00, 215.90)	(448.9, 650.00, 895.4)	(0.79, 0.84, 0.90)	(190.30, 193.00, 196.80)	(235.70, 350.00, 513.40)	(0.87, 0.91, 0.95)
21	(175.50, 180.00, 186.70)	(110.50, 270.00, 498.10)	(0.87, 0.93, 0.98)	(197.80, 202.00, 207.20)	(558.8, 732.00, 928.6)	(0.79, 0.83, 0.87)	(181.30, 184.00, 187.60)	(231.80, 354.00, 517.00)	(0.87, 0.91, 0.95)
22	(167.30, 172.00, 176.80)	(105.60, 274.00, 467.00)	(0.88, 0.93, 0.98)	(196.30, 200.00, 205.00)	(716.5, 890.00, 1087.4)	(0.76, 0.80, 0.84)	(165.10, 172.00, 180.30)	(116.70, 274.00, 544.00)	(0.86, 0.93, 1.00)
23	(159.60, 162.00, 165.30)	(88.80, 216.00, 379.30)	(0.90, 0.94, 0.99)	(178.80, 186.00, 193.30)	(544.10, 768.00, 1023.30)	(0.77, 0.82, 0.88)	(161.10, 167.00, 174.60)	(149.50, 331.00, 593.20)	(0.85, 0.91, 0.98)
24	(154.00, 156.00, 157.80)	(126.20, 234.00, 364.60)	(0.90, 0.94, 0.98)	(169.30, 176.00, 183.50)	(484.50, 714.00, 981.40)	(0.78, 0.83, 0.89)	(159.60, 162.00, 165.30)	(240.70, 378.00, 544.60)	(0.86, 0.90, 0.94)
25	(154.00, 156.00, 157.80)	(247.10, 390.00, 522.40)	(0.87, 0.90, 0.94)	(172.70, 178.00, 185.50)	(723.50, 940.00, 1214.90)	(0.74, 0.79, 0.84)	(154.00, 156.00, 157.80)	(262.40, 390.00, 522.40)	(0.87, 0.90, 0.94)

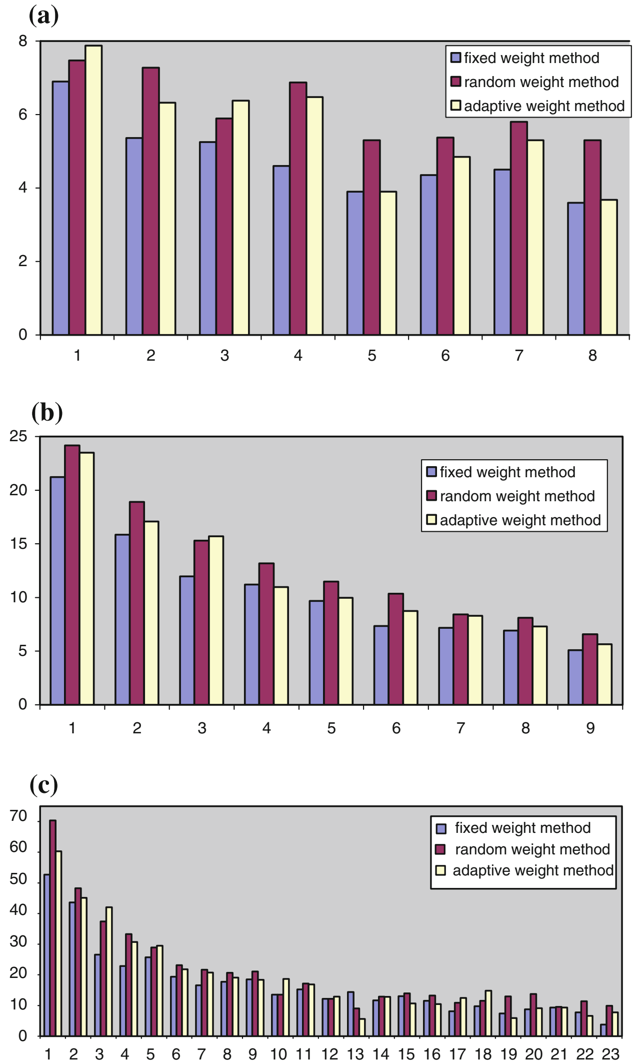

Fig. 5 The mean divergence (Eq. (9)) of the cycle times over the 10 runs of f-MOGA for the instances of **a** Sawyer, **b** Kilbridge, and **c** Tonge ALBP

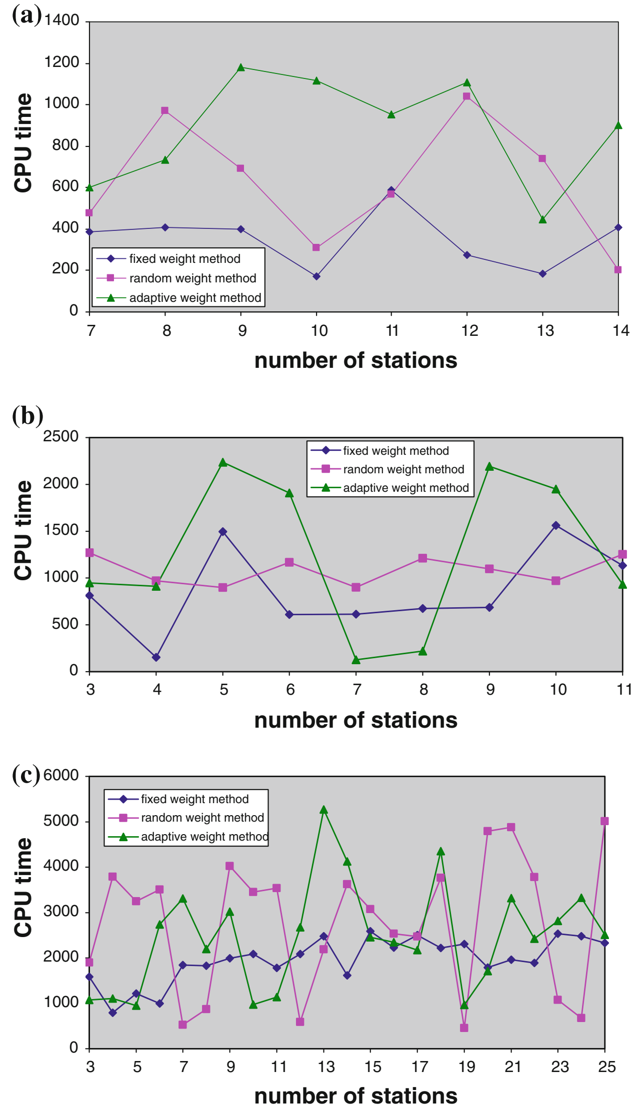

Fig. 6 The CPU time spent by the f-MOGA versus the number of stations for **a** Sawyer, **b** Kilbridge, and **c** Tonge ALBPs

Furthermore, the fuzzy cycle times obtained by f-MOGA are in average better using the fixed-weight method than those obtained using either the random-weight, or the adaptive-weight method. Figure 5 shows the divergence of the optimum cycle times (according to Eq. (9)) obtained by the three versions of f-MOGA over the examined ALBPs. It is clear from this diagram that, in most of the cases the fixed-weight method performs better than the other two methods (although the difference in the generated cycle times are not very large).

Figure 6 shows the CPU times (averaged over the 10 runs) spent by the three versions of f-MOGA over the examined ALBPs.The diagrams confirm that the three weighted-methods have similar performance in terms of running times. As one can see from Fig. 6a, f-MOGA+fixed-weight method is clearly faster than the other two approaches especially for small size problems (lower curve in Fig. 6a). This observation is not clear for the larger size problems (see Fig. 6b, c). However, the average running duration spent by f-MOGA+fixed-weighted method is still quite well with regard to the running durations of the other two approaches.

Conclusions

In this paper, a novel fuzzy extension of the simple assembly line balancing problem of type 2 (SALBP-2) has been proposed. The fuzzy job processing times reflect the uncertainty, variability and imprecision with which real-world production systems are afflicted. The jobs processing times are formulated by triangular fuzzy membership functions. A new multi-objective GA (MOGA) is introduced for solving the fuzzy SALBP-2 with objectives: to minimize the fuzzy cycle time, the fuzzy balance delay time, and the fuzzy smoothness index of the line.

The total fuzzy cost function is formulated as the weighted-sum of multiple fuzzy objectives. Three different methods for computing the weights in the fuzzy cost function were studied namely, fixed-, random- and adaptive-weight method, respectively. The influences of these methods on the performance of the proposed MOGA were examined over known test beds. The results obtained showed that the use of a fixed-weight method within the proposed approach exhibited superiority over the other two methods (with regard to both the quality of the solutions obtained and speed of convergence). The experimental results verify that the proposed MOGA is a powerful tool for solving fuzzy scheduling problems.

This work is limited in the single-model ALBP, however, it represents a good start point for further studies focused on more difficult ALBPs such as the mixed-model ALBP with fuzzy job processing times. This problem is much more complex than SALBP since, the attempt is to manufacture different models (versions) of the same basic product in the same line (e.g. PCs with or without DVD drive, with the graphics card on the mother board or not, etc.) in arbitrarily intermixed sequence. A first idea is to address the feasibility fuzzy mixed-model ALBP. That is, given the cycle time $\boldsymbol{c}$ and the number $\boldsymbol{m}$ of the workstations in the line determine whether, or not a feasible mixed-model assignment with $\boldsymbol{m}$ stations exists.

References

Anderson, E. J., & Ferris, M. C. (1994). Genetic algorithms for combinatorial optimization: the assembly line balancing problem. *INFORMS Journal on Computing, 6*, 161–173.

Bäck, T. (1996). *Evolutionary algorithms in theory and practice*. New York, NY: Oxford University Press.

Baudin, M. (2002). *Lean assembly: The nuts and bolts of making assembly operations flow, productivity*. New York: Productivity Press.

Baybars, I. (1986). A survey of exact algorithms for the simple assembly line balancing problem. *Management Science, 32*, 909–932.

Baykasoglu, A. (2006). Multi-rule multi-objective simulated annealing algorithm for straight and U type assembly line balancing problems. *Journal of Intelligent Manufacturing, 17*, 217–232.

Becker, C., & Scholl, A. (2006). A survey on problems and methods in generalized assembly line balancing. *European Journal of Operational Research, 168*(3), 694–715.

Brudaru, O., & Valmar, B. (2004). Genetic Algorithm with embryonic chromosomes for assembly line balancing with fuzzy processing times. In *8th international research/expert conference trends in the development of machinery and associated technology*, TMT 2004. Neum, Bosnia and Herzegovina.

Chiang, W. C. (1998). The application of a tabu search metaheuristic to the assembly line balancing problem. *Annals of Operations Research, 77*, 209–227.

Erel, E., & Sarin, S. (1998). A survey of the assembly line balancing procedures. *Production Planning and Control, 9*(5), 414–434.

Gen, M., & Cheng, R. (2000). *Genetic algorithms and engineering optimisation*. New York, NY: Wiley-Interscience.

Gen, M., Tsujimura, Y., & Li, Y. (1996). Fuzzy assembly line balancing using genetic algorithms. *Computers and Industrial Engineering, 31*(3/4), 631–634.

Glover, F. (1989). Tabu-search-Part I. *ORSA Journal Computing, 1*(3), 190–206.

Glover, F. (1990). Tabu-search-Part II. *ORSA Journal Computing, 2*(1), 4–32.

Goldberg, D. E. (1989). *Genetic algorithm in search, optimization and machine learning*. Reading, Massachusetts: Addison Wesley.

Heinrici, A., et al. (1994). A comparison between simulated annealing and tabu search with an example from the production planning. In H. Dyckhoff (Ed.), *Operations research proceedings 1993* (pp. 498–503). Berlin: Springer.

Holland, J. H. (1975). *Adaption in natural and artificial systems*. Ann Arbor, MI: University of Michigan Press.

Kaufmann, A., & Gupta, M. M. (1985). *Introduction to fuzzy arithmetic*. New York: Van Nostrand Reinhold.

Kim, Y. K., Kim, Y. J., & Kim, Y. (1996). Genetic algorithms for assembly line balancing with various objectives. *Computers and Industrial Engineering, 30*(3), 397–409.

Kirkpatrick, S., Gelatt, C. D., Jr., & Vecchi, M. P. (1983). Optimization by simulated annealing. *Science, 220*, 671–680.

Michalewitz, Z. (1996). *Genetic algorithms + data structures = evolution programs* (3rd ed.). Berlin: Springer.

Murata, T., Ishibuchi, H., & Tanaka, H. (1996). Multi-objective genetic algorithms and its application to flowshop scheduling. *Computers and Industrial Engineering, 30*(4), 957–968.

Nearchou, A. C. (2008). Multi-objective balancing of assembly lines by population heuristics. *International Journal of Production Research, 46*(8), 2275–2297.

Oman, S., & Cunningham, P. (2001). Using case retrieval to seed genetic algorithms. *International Journal of Computational Intelligence and Applications, 1*(1), 71–82.

Ozcan, U., & Toklu, B. (2009). A new hybrid improvement heuristic approach to simple straight and U-type assembly line balancing problems. *Journal of Intelligent Manufacturing, 20*, 123–136.

Rekiek, B., De Lit, P., Pellichero, F., L'Englise, T., Fouda, P., & Falkenauer, E., et al. (2001). A multiple objective grouping genetic algorithm for assembly line design. *Journal of Intelligent Manufacturing, 12*, 467–485.

Sabuncuoglu, I., Erel, E., & Tanyer, M. (2000). Assembly line balancing using genetic algorithms. *Journal of Intelligent Manufacturing, 11*, 295–310.

Scholl, A. (1999). *Balancing and sequencing of assembly lines*. Heidelberg, Germany: Physica-Verlag.

Scholl, A., & Becker, C. (2006). State of the art exact and heuristic solution procedures for simple assembly line balancing. *European Journal of Operational Research, 168*(3), 666–693.

Scholl, A., & Voß, S. (1996). Simple assembly line balancing—Heuristic approaches. *J Heuristics, 2*, 217–244.

Tasan, S. O., & Tunali, S. (2008). A review of the current applications of genetic algorithms in assembly line balancing. *Journal of Intelligent Manufacturing, 19*(1), 49–69.

Tsujimura, Y., Gen, M., & Kubota, E. (1995). Solving fuzzy assembly-line balancing problem with genetic algorithms. *Computers and Industrial Engineering, 29*(1–4), 543–547.

Watanabe, T., Hashimoto, Y., Nishikawa, I., & Tokumaru, H. (1995). Line balancing using a genetic evolution model. *Control Engineering Practice, 3*, 60–76.

Zhang, W., Gen, M. (2009). An efficient multiobjective genetic algorithm for mixed-model assembly line balancing problem considering demand ratio-based cycle time. *Journal of Intelligent Manufacturing.*, (available on-line) doi:10.1007/s10845-009-0295-5.