Int J Adv Manuf Technol (2007) 34:748–761
DOI 10.1007/s00170-006-0636-x

ORIGINAL ARTICLE

Work-in-process scheduling by evolutionary tuned fuzzy controllers

N. Tsourveloudis · L. Doitsidis · S. Ioannidis

Received: 9 February 2005 / Accepted: 18 April 2006 / Published online: 18 August 2006

Abstract In this paper, an evolutionary algorithm (EA) strategy for the optimization of generic work-in-process (WIP) scheduling fuzzy controllers is presented. The EA strategy is used to tune a set of fuzzy control modules that are used for distributed and supervisory WIP scheduling. The distributed controllers objective is to control the rate in each production stage in a way that satisfies the demand for final products while reducing WIP within the production system. The EA identifies those sets of parameters for which the fuzzy controller performs optimal with respect to WIP and backlog minimization. The proposed EA strategy is compared with known heuristically tuned distributed and supervised fuzzy control approaches. Extensive simulation results show that the EA strategy significantly improves the system's performance.

Keywords Work-in-process · Backlog fuzzy control · Evolutionary algorithms · Production systems

N. Tsourveloudis (✉) · L. Doitsidis
Department of Production Engineering and Management,
Technical University of Crete,
University Campus,
73100 Chania, Greece
e-mail: nikost@dpem.tuc.gr

L. Doitsidis
e-mail: ldoitsidis@dpem.tuc.gr

S. Ioannidis
Department of Mathematics, University of the Aegean,
83200 Karlovasi, Samos, Greece
e-mail: efioan@aegean.gr

Nomenclature

M_i	Machine i
s_i	State of M_i
p_i	Failure rate of M_i
rr_i	Repair rate of M_i
μ_i	Maximum processing rate of M_i
r_i	Procesing rate of M_i
PR_i	Cumulative production of M_i
$B_{j,i}$	Buffer receiving items from M_j and supplying M_i
$BC_{j,i}$	Buffer capacity of $B_{j,i}$
$b_{j,i}$	Buffer level of $B_{j,i}$
x_i	Production surplus of M_i
x_e	Surplus of the end product
e_x	Error of the end product surplus
mx_e	Mean surplus of the end product
WIP	Work-in-process (including the end product buffer level)
e_w	Relative error of WIP
u_c	Upper surplus bound correction factor
l_c	Lower surplus bound correction factor
d	Demand rate of end products
D	Cumulative demand of end products
C	Mean production cost
c_I	Inventory unit cost
c_b	Backlog unit cost
BL	Mean backlog
LY	Linguistic value of variable y
Y	Term set of linguistic value LY
$\mu_Y(y)$	Membership function of variable y
$\bar{y}$	Mean value of variable y
$y(t)$	Value of variable y at time t
$F(y)$	Fitness function of variable y
HDF	Heuristic distributed fuzzy approach
EDF	Evolutionary distributed fuzzy approach

HSF Heuristic supervisory fuzzy approach
ESF Evolutionary supervisory fuzzy approach

1 Introduction

The increased need for speedy and punctual delivery of products and goods has placed an increased emphasis on production scheduling methods. Traditionally, production scheduling manages the flow of materials or components through the manufacturing system. The random events occurring in a manufacturing system makes realistic scheduling a difficult task. Recent advances in manufacturing methods and concepts such as FMS, JIT, agile and lean manufacturing, have resulted in improving processes but have also led to changes in manufacturing management. In a highly changing demand environment, the accumulated inventories are less desirable than ever. Further, the in-process inventories, widely known as work-in-process (WIP), should stay as small as possible (for various reasons reported in [1] and elsewhere [2]).

The work-in-process inventory is measured by the number of unfinished parts in the buffers throughout the manufacturing system. The important question in WIP management is: *what is the minimum necessary WIP*? The answer, which is not straightforward, is that WIP is highly associated with the fluctuations of demand. WIP is accumulated when the actual production rate is higher than the demand. However, when WIP is very low, unpredicted phenomena, such as machine failures, may lead the actual production behind the demand and thus delay deliveries and cause unsatisfied customers. Obviously, product demands of a constant level and pattern make the scheduling task easier than randomly changing demands.

Control policies that tend to keep WIP at low levels have drawn a great deal of attention from researchers and practitioners [2, 3]. Recently, artificial intelligence-based methodologies for the WIP control of realistic (in terms of modelling assumptions) manufacturing systems have been presented [4–6].

In [4], a distributed fuzzy control methodology for single and multiple part-type production lines and networks was introduced. The control objective was to keep the WIP and cycle time as low as possible, and at the same time to maintain high machine utilization. In contrast to the traditional produce-at-capacity approach, according to which the system always operates at its maximum capacity, the production rate was controlled in each production stage in a way that eliminates extreme events of idle periods due to machine starving or blocking. The controllers presented although appear to be better compared to traditional and surplus-based policies were heuristically tuned and therefore cannot assure optimal behavior.

A supervisory WIP control scheme that keeps the in-process inventory and backlog at low levels was introduced by Ioannidis et al. in [5]. However, this approach has also not adopted a systematic methodology that ensures optimal design of the in–process inventory controllers.

In this paper, we present an evolutionary algorithm (EA) strategy for the optimization of generic WIP scheduling fuzzy controllers introduced in [4] and [5]. The scheduling problem objective is to control the production rate in a way that satisfies the (random) demand for final products while keeping minimum WIP within the production system. During the evolution, the EA identifies those set of parameters for which the fuzzy controller performs optimal with respect to WIP minimization for several demand patterns.

The rest of the paper is organized as follows. Section 2 describes the heuristic fuzzy control architectures. Section 3 presents the evolutionary algorithm, which is applied for the improvement of the fuzzy control schemes. Section 4 describes experimental results for production lines and networks. Issues for discussion and suggestions for further development are presented in Section 5.

2 Fuzzy production scheduling

A production system can be viewed as a network of machines and buffers. Items are received at each machine and wait for the next operation in a buffer with finite capacity. The machines can break down in a random order and sometimes can be incapable of producing more parts because of starvation and/or blocking phenomena. Due to a failed machine with operational neighbors, the level of the downstream buffer decreases, while the upstream increases. If the repair time is big enough, then the broken machine will either block the next station or starve the previous one. This effect will propagate throughout the system.

Clearly, production scheduling of realistic manufacturing plants must satisfy multiple conflicting criteria and also cope with the dynamic nature of such environments. Fuzzy logic offers the mathematical framework that allows for simple knowledge representations of the production control/scheduling principles in terms of IF-THEN rules.

In fuzzy logic controllers (FLCs), the control policy is described by linguistic IF-THEN rules, which model the relationship between control inputs and outputs with appropriate mathematical representation. A rule antecedent (IF-part) describes conditions under which the rule is applicable and forms the composition of the inputs. The consequent (THEN-part) gives the response or conclusion that should be taken under these conditions. A two-input (antecedent) rule of the Mamdani type has the form [7]:

IF X is A AND Y is B THEN Z is C,

where X, Y are the input and Z is the output variable, and A, B and C their linguistic variations, respectively, that are fuzzy sets with certain membership functions. The crisp control action is obtained through a defuzzification method, which, in most applications, calculates the centroid of the output fuzzy set.

2.1 Distributed fuzzy scheduling

The advantage of the distributed FLCs is that they are computationally simple and therefore facilitate application to real-time control/scheduling. In the distributed fuzzy scheduling system presented in [4], three basic subsystems have been introduced, namely *transfer line*, *assembly* and *disassembly* module. The majority of the real production networks can be decomposed into these basic subsystems. Each subsystem can be seen as a distributed fuzzy logic controller with input and output variables summarized in Table 1.

The control objective of the distributed scheduling approach, as earlier stated, is to satisfy the demand and, at the same time, to keep WIP as low as possible. This is attempted by regulating the processing rate r_i at every time instant. The expert knowledge that describes the control objective can be summarized in the following statements:

If the surplus level is satisfactory, then try to prevent starving or blocking by increasing or decreasing the production rate accordingly.

If the surplus is not satisfactory, that is, either too low or too high, then produce at maximum or zero rate, respectively.

The above knowledge may be more formally represented, for each one of the control modules of Table 1, by fuzzy rules of the following form:

Transfer Line Rule : IF $b_{j,i}$ is $LB^{(k)}$ AND $b_{i,l}$ is $LB^{(k)}$ AND s_i is $LS_i^{(k)}$ AND x_i is $LX^{(k)}$ THEN r_i is $LR_i^{(k)}$

Assembly node Rule : IF $b_{j,i}$ is $LB^{(k)}$ AND...AND $b_{k,i}$ is $LB^{(k)}$ AND $b_{i,l}$ is $LB^{(k)}$ AND s_i is $LS_i^{(k)}$ AND x_i is $LX^{(k)}$, THEN r_i is $LR_i^{(k)}$

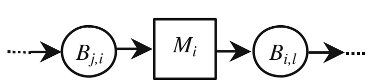
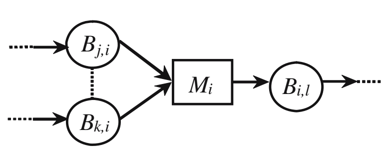
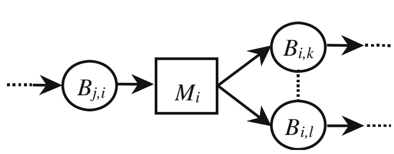
Disassembly node Rule : IF $b_{j,i}$ is $LB^{(k)}$ AND $b_{i,k}$ is $LB^{(k)}$ AND...AND $b_{i,l}$ is $LB^{(k)}$ AND s_i is $LS_i^{(k)}$ AND x_i is $LX^{(k)}$, THEN r_i is $LR_i^{(k)}$

where k is the rule number, i is the number of machine or workstation, LB is a linguistic value of the variable *buffer levelb* with term set B={*Empty*, *Almost Empty*, *OK*, *Almost Full*, *Full*}, s_i denotes the state of machine i, which can be either 1 (operative) or 0 (stopped); consequently S={0, 1}. LX represents the value that surplus x takes and it is chosen from the term set X= {*Negative*, *OK*, *Positive*}. The *production rate* r takes linguistic values LR from the term set R= {*Zero*, *Low*, *Almost Low*, *Normal*, *Almost High*, *High*}. The processing rate r_i of each machine at every time instant is

$$r_i^{'} = \mathbf{f}_{\mathrm{IS}}\left(b_{j,i},\ b_{i,l},\ x_i,\ s_i\right) = \begin{cases} 0 & \text{if } s_i = 0 \\ \frac{\sum r_i \mu_R^*(r_i)}{\sum \mu_R^*(r_i)} & \text{if } s_i = 1, \end{cases} \quad (1)$$

where, $\mathbf{f}_{\mathrm{IS}}\left(b_{j,i},\ b_{i,l},\ x_i,\ s_i\right)$ represents a fuzzy inference system [7, 8] that takes as inputs the level $b_{j,i}$ of the upstream

Table 1 Control modules

Module	Schema	Inputs	Output
Line		b_{ji} buffer level of buffer $B_{j,i}$ b_{il} buffer level of buffer $B_{i,l}$ s_i state of machine M_i x_i production surplus of M_i	
Assembly		b_{ji} buffer level of buffer $B_{j,i}$ b_{ki} buffer level of buffer $B_{k,i}$ b_{il} buffer level of buffer $B_{i,l}$ s_i state of machine M_i x_i production surplus of M_i	r_i processing rate of M_i
Disassembly		b_{ji} buffer level of buffer $B_{j,i}$ b_{ik} buffer level of buffer $B_{i,k}$ b_{il} buffer level of buffer $B_{i,l}$ s_i state of machine M_i x_i production surplus of M_i	

buffer, the downstream buffer level $b_{i,l}$, x_i is the surplus (cumulative production minus demand) and s_i is a non-fuzzy variable denoting the state of the machine, which can be either 1 (operative) or 0 (stopped). In Eq. (1), $\mu_R^*(r_i)$is the membership function of the aggregated production rate, which is given by

$$\mu_R^*(r_i) = \max_{b_{j,i},b_{i,l},x_i} \min\left[\mu_{AND}^*\left(b_{j,i},\ b_{i,l},\ x_i\right), \mu_{FR^{(k)}}\left(b_{j,i},\ b_{i,l},\ x_i,\ r_i\right)\right], \quad (2)$$

where $\mu_{AND}^*\left(b_{j,i},\ b_{i,l},\ x_i\right)$ is the membership function of the conjunction of the inputs and $\mu_{FR^{(k)}}\left(b_{j,i},\ b_{i,l},\ x_i,\ r_i\right)$ is the membership function of the *k*-th activated rule. That is

$$\mu_{AND}^*\left(b_{j,i},\ b_{i,l},\ x_i\right) = \mu_B^*\left(b_{j,i}\right) \wedge \mu_B^*\left(b_{i,l}\right) \wedge \mu_X^*(x_i), \quad (3)$$

and

$$\mu_{FR^{(k)}}\left(b_{j,i},\ b_{i,l},\ x_i,\ r_i\right) = \mathbf{f}_{\rightarrow}\left[\mu_{LB^{(k)}}\left(b_{j,i}\right),\ \mu_{LB^{(k)}}\left(b_{i,l}\right),\ \mu_{LX^{(k)}}(x_i),\ \mu_{LR^{(k)}}(r_i)\right]. \quad (4)$$

In Eqs. (3 and 4), $\mu_B^*\left(b_{j,i}\right)$ and $\mu_B^*\left(b_{i,l}\right)$ are the membership functions (MFs) of the actual upstream and downstream buffer levels and $\mu_X^*(x_i)$ is the membership function of production surplus.

2.2 Supervised fuzzy scheduling

In control systems literature, a supervisory controller utilizes macroscopic data of higher hierarchies to adjust the overall system's behavior. Potentially, this may happen by modifying the lower level controllers in a way to ultimately achieve desired specifications. The supervisory controller's task, introduced in [5] and its optimization discussed in the next paragraph, is the tuning of the distributed fuzzy controllers presented in the previous paragraph, in a way that improves certain performance measures without causing a dramatic change in the control architecture. Therefore, the overall scheduling approach remains modular since the production control modules are not modified but simply tuned by the additional supervisory controller.

In the supervisory scheduling scheme it is assumed that the demand and the cumulative production are known. This is important for the production surplus monitoring and control and, consequently, for scheduling decisions based on production surplus. The expert knowledge that describes the supervisory control objective builds on the assumption that adaptive surplus bounds may improve the production systems performance and can be summarized in the following statements:

- If the upper surplus bound is reduced, there is an immediate reduction of WIP.
- If the upper surplus bound is increased, there is an increase of WIP and the total production rate, leading to a small reduction of backlog.
- If the lower surplus bound is increased, a substantial reduction of backlog and an increase in WIP is achieved.
- If there is a reduction of lower surplus bound as a result we have a deterioration of backlog with an improvement of WIP.

Surplus bounds are decided by the output of IF-THEN rules of the following form:

IF mx_e is $LMX^{(k)}$ AND e_x is $LE_x^{(k)}$ AND e_w is $LE_w^{(k)}$ THEN u_c is $LU_c^{(k)}$ and l_c is $LL_c^{(k)}$,

where, k is the rule number, mx_e is the mean surplus of the end product, *LMX* is a linguistic value of the mx_e with term set *MX*={*Negative Big*, *Negative Small*, *Zero*, *Positive Small*, *Positive Big*}, e_x denotes the error of end product surplus (=the difference between surplus x_e and the lower bound of surplus), with linguistic value term set E_x={*Negative*, *Zero*, *Positive*}. The relative deviation of *WIP* is denoted e_w and LE_w is the corresponding linguistic value is chosen from the term set E_w={*Negative*, *Zero*, *Positive*}. The upper surplus bound correction factor u_c takes linguistic values LU_c from the term set U_c={*Negative*, *Negative Zero*, *Zero*, *Positive Zero*, *Positive*} and the lower surplus bound correction factor l_c takes linguistic values LL_c from the term set L_c={*Negative*, *Negative Zero*, *Zero*, *Positive Zero*, *Positive*}.

The crisp arithmetic values, u_c^* and l_c^*, of the corrections of the upper and lower surplus bounds, respectively, are given by the following defuzzification formulas:

$$u_c^* = \frac{\sum u_c \cdot \mu_{U_c}^*(u_c)}{\sum \mu_{U_c}^*(u_c)} \quad (5a)$$

$$l_c^* = \frac{\sum l_c \cdot \mu_{L_c}^*(l_c)}{\sum \mu_{l_c}^*(u_c)} \quad (5b)$$

where $\mu_{U_c}^*(u_c)$ and $\mu_{L_c}^*(l_c)$ are the membership functions of the upper and lower surplus bounds, respectively. These membership functions represent the aggregated outcome of the fuzzy inference procedure. The correct selection of input and output membership functions characterizes the performance of the overall scheduling task.

2.3 Membership function selection

The essential part of every fuzzy controller is the knowledge acquisition and the representation of the extracted knowledge with certain fuzzy sets/membership functions. Usually the designer observes and/or interviews the human operator of the plant in order to derive the control logic and the important for the process variables.

Membership functions (MFs) represent the uncertainty modeled with fuzzy sets by establishing a connection between linguistic terms (such as *low*, *negative*, *high,* etc.) and precise numerical values of variables in the physical system.

The correct choice of the MFs is by no means trivial but plays a crucial role in the success of an application. Consequently, the selection of membership functions, if not based on a systematic optimization procedure, cannot guarantee a minimum WIP level. This is the main drawback of the heuristic selection of MFs in case of known (or almost known) demand patterns. The evolutionary algorithm developed and explained in the next section creates MFs that fit best to scheduling objectives. In this context, the design of the fuzzy controllers (distributed or supervisory) can be regarded as an optimization problem in which the set of possible MFs constitutes the search space.

3 Evolutionary fuzzy scheduling

Evolutionary (EAs) and genetic algorithms (GAs) are inspired by nature in the sense of the fundamental concept of the survival of the fittest as being encountered in selection mechanisms among species. They are seeking optimal or near-optimal solutions in large and complex search spaces and therefore have been successfully applied to a variety of scheduling problems with broad applicability to manufacturing systems [9]. The basic idea in evolutionary schemes is to maintain a population of chromosomes that evolves over time through a process of competition and controlled variation.

The use of evolving genetic structures for the production scheduling problem, has recently gained a lot of acceptance for the automated and optimal design of fuzzy logic systems [10]. The applications of evolutionary techniques in fuzzy logic controllers are concentrated in two main subjects: (a) the optimization of membership functions and (b) the automatic learning and extraction of fuzzy rules (details in [11]). Here, we consider the application of an evolutionary algorithm for the optimal selection of MFs. In the literature, some authors argue that membership functions and fuzzy rules are co-dependent and should be evolved simultaneously [12]. However, the scheduling knowledge and the corresponding rules used here are readily available and relatively easy to understand and therefore it is not necessary to use EAs to evolve the fuzzy rules. In general, is it better to say that the right use of evolutionary strategies is situation-dependent.

The membership functions defined in the previous paragraphs are used to construct the *chromosome*. The basic idea is to represent the complete set of membership functions by an individual (chromosome) and to evolve shape and location of the MFs. This is shown in Fig. 1 for the case of trapezoidal and triangular MFs. An initial population is then derived from the first chromosome by repeated application of the mutation operator. The objective is to optimize a performance measure which in the EAs context is called *fitness function*. In each generation, the fitness of every chromosome is first evaluated based on the performance of the production network system, which is controlled through the membership functions represented in the chromosome. A specified percentage of the better-fit chromosomes are retained for the next generation. Then parents are selected repeatedly from the current generation of chromosomes, and new chromosomes are generated from these parents. One generation ends when the number of chromosomes for the next generation has reached the quota. This process is repeated for a pre-selected number of generations. The pseudo-code that describes the procedure is presented in Fig. 2.

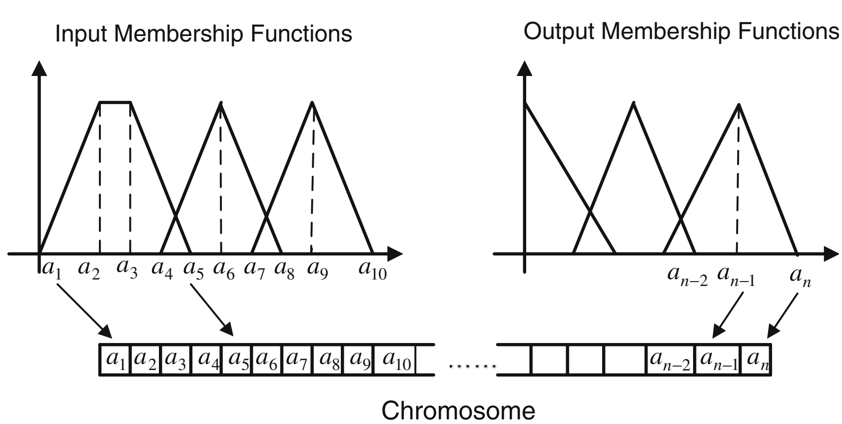


Fig. 1 Chromosome created by the membership functions

```
Initialization (Creation of the initial population)
  For i = 1 to number of generations
    For j = 1 to number of individuals
      Create the membership functions for the individual j
      Run the simulation of the production network
      Evaluate fitness of the individual j
    End j
  Rank the individuals based on their fitness function
  Select the fittest and mutate them
  End i
```

Fig. 2 Pseudo code of the evolution procedure

4 Distributed evolutionary scheduling

The structure of the distributed fuzzy logic controllers as far as it concerns the rule base and the linguistic variables remains the same with those described in Section 2. The controllers used for training have randomly created membership functions.

The initial population consists of individuals that have the same initial chromosome which contains the points a_i, (i=*1*,..., n) that define the membership functions of the inputs and the output of the distributed controllers. In the case of more than one controller, the chromosome consists of the points that define all membership functions of these controllers. As previously mentioned, the membership functions, which correspond to the linguistic variables, are randomly created in the beginning of the evolution process. The evolutionary algorithm maintains a population of individuals in each generation/iteration. Individuals represent a different set of distributed fuzzy controllers. In every generation the individuals are sorted from the best to the worst based on their fitness score. The fitness function used considers the demand in the system and the cumulative production of the system. The fitness $F(x_i)$, of each individual x_i is selected as:

$$F(x_i) = \left[\sum_{j=1}^{N} \left(D(t_j) - PR(t_j) \right)^2 \right]^{-1} \tag{6}$$

where, t is the current simulation time, T is the total simulation time and $D(t)$ is the overall demand and $PR(t)$ is the cumulative production of the system. The architecture of the distributed evolution scheme is presented in Fig. 3.

The best individual is considered to be the one with the biggest fitness. The fittest individuals are selected and they undergo mutations. The fittest controllers and their mutated offsprings are forming the new population. After some generations, the algorithm converges and the best individuals represent near-optimal solutions. The production systems were designed and implemented into *Simulink*, while the code for the evolution process was designed into MATLAB. After the evolution process the shape of the membership functions is

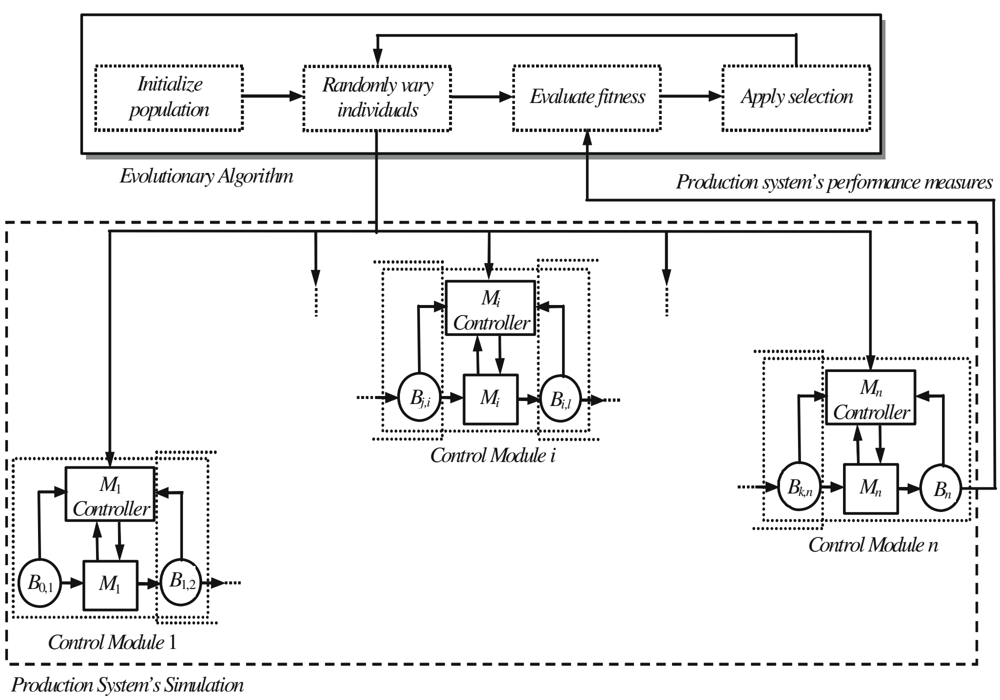


Fig. 3 Distributed fuzzy control evolution concept

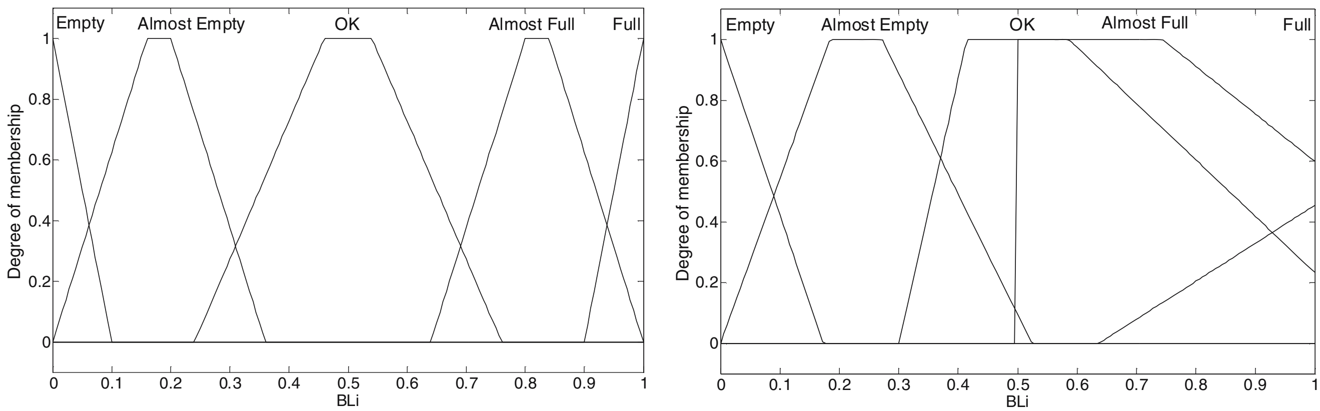


Fig. 4 Input variable *buffer level* **a** before, and **b** after the evolution process

altered. An example of the membership functions change for the input variable *buffer level* is presented in Fig. 4.

5 Supervisory evolutionary scheduling

The evolutionary optimization procedure shown in Fig. 2 was used for the supervisory controller. In the lower control level were used the heuristic fuzzy distributed controllers introduced in [4]. The parameters were chosen as in the distributed case, that is, the membership functions of input and output variables presented in Section 2. The population number is 40 and the mutation rate is selected 0.1. From the overall population, the 20 fittest individuals are qualified for the next generation while the rest are replaced by mutation of the fittest. Each individual is evaluated by the results of a simulation run of 200 time units. The architecture of the supervisor evolution scheme is shown in Fig. 5.

The fitness function in the supervisory approach case was chosen to be the following:

$$F = \frac{1}{c_I\overline{WIP} + c_b\overline{BL}} \tag{7}$$

where, $\overline{WIP}$ and $\overline{BL}$ are the mean work-in-process and mean backlog, respectively. The c_I, c_b are weighting factors that represent the unit costs of inventory and backlog, respectively. By taking into account these costs in the fitness function, we may adjust the importance of $\overline{WIP}$ and $\overline{BL}$. It can be seen in what follows that when the contribution of *WIP* to the production cost is much greater than the contribution of backlog, the heuristic selection of the MFs may outperform the evolutionary one.

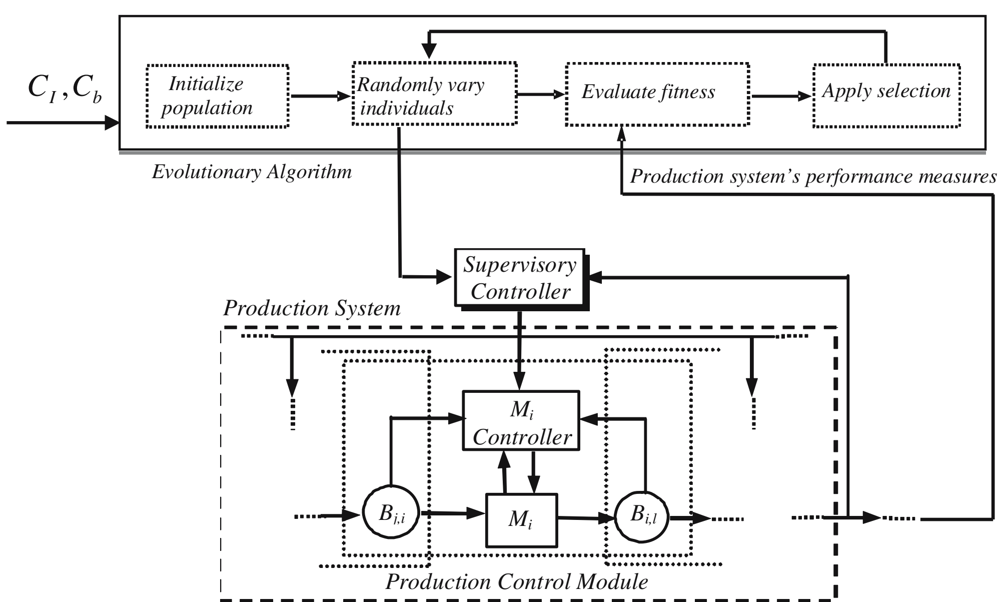


Fig. 5 Supervisory control evolution concept

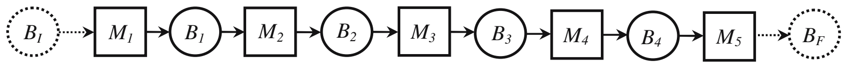


Fig. 6 Test case 1: Production line

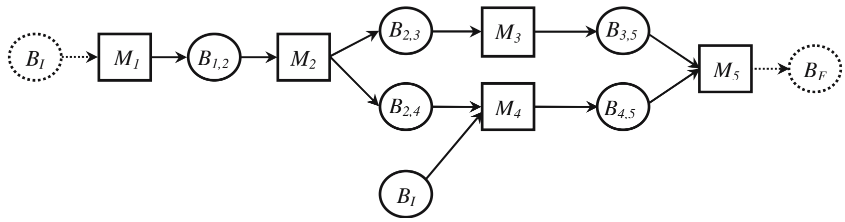


Fig. 7 Test case 2: Production network

6 Simulation testing and results

We have used the evolutionary algorithm presented in the previous section in order to optimize the performance of the unsupervised/distributed and the supervised production control schemes. The proposed evolutionary fuzzy approaches are tested and compared with the heuristic approaches introduced in [4] and [5]. Three test cases are considered. A single-part-type production line (Fig. 6), a single-part-type production network (Fig. 7) and a real olive pomace oil processing plant (Fig. 8). We assume that the flow of parts within the system is continuous. In the continuous-flow simulation, the discrete production is approximated by the production of a liquid product [13]. The assumptions made for all simulations are the following:

– Machines fail randomly with a failure rate p_i.
– Machines are repaired randomly with rate rr_i. Unlimited repair personnel is assumed.
– Time to failure and time to repair are exponentially distributed.
– Demand is either constant or stochastic with rate d. In the stochastic case, it follows the Poisson distribution.
– All machines operate at known (but not necessarily equal) rates. Each machine produces in a rate $r_i \leq \mu_i$, where μ_i is the maximum processing rate of machine M_i.
– The initial buffers are infinite sources of raw material and consequently the initial machines are never starved.
– Buffers between adjacent machines M_i, M_j have finite capacities.
– Set-up times or transportation times are negligible or are included in the processing times.

The buffer levels at any time instant is given by:

$$b_{j,i}(t_{k+1}) = b_{j,i}(t_k) + \left[r_j(t_k) - r_i(t_k)\right](t_{k+1} + -t_k), \tag{8}$$

where t_k ,t_{k+1}, r_i are the times when control actions (changes in processing rates), happen. The cumulative production of a machine M_i is

$$PR_{i(t_{k+1})} = PR_{i(t_k)} + r_i(t_k)(t_{k+1} - t_k), \tag{9}$$

The mean machine rate $\overline{r_i}$ is given by

$$\overline{r}_i = \frac{PR_i(T)}{T}, \tag{10}$$

where T is the total simulation time.

6.1 Distributed evolutionary fuzzy approach

The developed evolutionary algorithm was used for the parameters selection of the heuristic distributed fuzzy controllers presented in Section 2.1. The fitness function, which was shown in Eq. (6) can be rewritten as:

$$F = \frac{1}{\sum_{i=1}^{N} [x(t_i)]^2}, \tag{11}$$

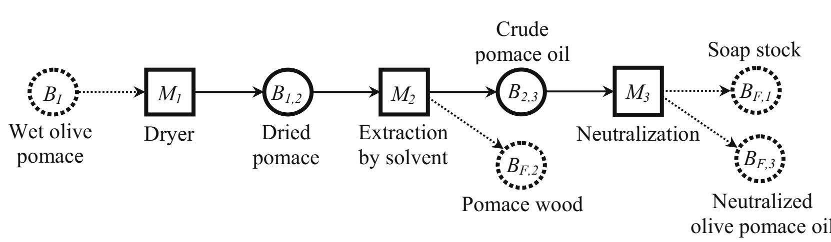


Fig. 8 Test case 3: Production system of an olive pomace oil processing plant

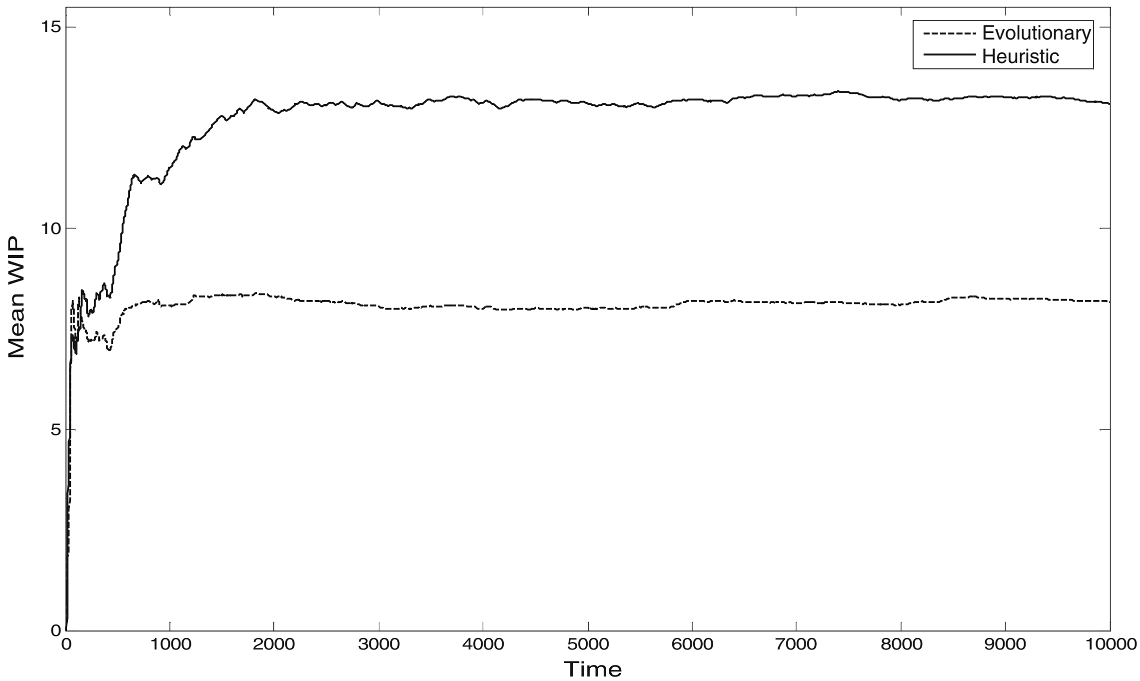


Fig. 9 Evolution of $\overline{WIP}$ in the production line of test case 1 with stochastic demand (d=1)

where $x(t_i)$ is the production surplus of the production system at time t_i.. This fitness function was chosen in order to keep the surplus as close to zero as possible. A systems surplus close to zero suggests that the system satisfy demand by keeping backlog in low levels while the finished items inventory level is also sustained low. The performance of the evolutionary distributed fuzzy approach (EDF) is compared with the heuristic distributed fuzzy control scheme (HDF). Two test cases are examined: a single-part-type production line (Fig. 6) and a single-part-type production network (Fig. 7).

6.1.1 Test case 1: single-part-type production lines

The EDF approach is first tested for the case of a production line presented in Fig. 6. The production line under consideration consists of five machines producing one product type. The failure and repair rates are equal for all machines. The repair rates are rr_i=0.5 and the failure rates are p_i=0.1. The processing rates are also equal for all machines and are equal to μ_i=2 (i=1,...,5).

Table 2 Comparative results for test case 1

Demand		HDF $\overline{WIP}$	$\overline{BL}$	EDF $\overline{WIP}$	$\overline{BL}$
Constant	1	11.492	1.72	6.371	1.567
Stochastic	0.5	19.393	0.057	6.417	0.438
	1	12.719	2.496	8.07	2.427

In Fig. 9 the evolution of $\overline{WIP}$ for both evolutionary and heuristic systems in a simulation run of 10,000 time units is presented. Figure 10 shows the evolution of mean backlog $\overline{BL}$ for both systems. Figure 11 presents the evolution of the fitness functions of the two best individuals for the ESF approach. Comparative results for the $\overline{WIP}$ and $\overline{BL}$ for various demand patterns are shown in Table 2. All buffer capacities are equal to BC_i=10.

In order to comprehend better the significance of the results, a cost analysis is carried out. The production cost associated with the proposed control architecture consists of inventory and backlog costs. Inventory costs are due to the capital invested for the purchase of raw material and the handling of material during the production process. It is assumed that the inventory cost is independent from the

Table 3 Cost analysis for test case 1 for stochastic demand

Demand	c_I	c_b	Cost C	
			HDF	EDF
0.5	0.99	0.01	19.2	6.357
	0.75	0.25	14.56	4.922
	0.5	0.5	9.725	3.428
	0.25	0.75	4.891	1.933
	0.01	0.99	0.25	0.498
1	0.99	0.01	12.617	8.014
	0.75	0.25	10.163	6.659
	0.5	0.5	7.608	5.249
	0.25	0.75	5.052	3.834
	0.01	0.99	2.598	2.483

Table 4 Comparative results for test case 2

Demand		HDF		EDF	
		$\overline{WIP}$	$\overline{BL}$	$\overline{WIP}$	$\overline{BL}$
Constant	1	21.356	0.078	16.542	0.737
Stochastic	0.5	20.097	0.049	17.34	0.136
	1	20.496	0.087	10.046	0.673

stage of process. Thus, the mean production cost C is given by:

$$C = c_I\overline{WIP} + c_b\overline{BL}\,, \tag{12}$$

where c_I, c_b are the unit costs of inventory and backlog, respectively.

The cost analysis results for the production line examined in the test case for stochastic demand are presented in Table 3, where the production cost of the EDF control approach is compared with the HDF approach for various values of c_I and c_b.

6.1.2 Test case 2: single-part-type production networks

The second test case is the single-part production network presented in Fig. 7. The production system under consideration consists of five machines producing one part type. The failure and repair rates of all machines are equal. The repair rates are rr_i=0.5 and the failure rates are p_i=0.1. The processing rates are also equal for all machines and are equal to μ_i=5 (i=1,...,5). All buffer capacities are equal to BC_i=10. Comparative results for the $\overline{WIP}$ and $\overline{BL}$ for various demand patterns are shown in Table 4.

6.2 Supervised evolutionary fuzzy approach

The performance of the evolutionary supervised fuzzy (ESF) approach is compared with the heuristic supervised fuzzy control scheme (HSF) and the evolutionary distributed fuzzy control scheme. As in the case of the distributed evolutionary fuzzy, we have examined the two test cases of a single-part-type production line and a single-part-type production network presented in Figs. 6 and 7.

6.2.1 Test case 3: single-part-type production lines

Here, the ESF approach is tested for the case of the production line presented in Fig. 6. The values of the production line parameters are the same as in Section 6.1.1.

Comparative results for the $\overline{WIP}$, $\overline{BL}$ and production cost C, when c_I and c_b are equal to 0.5, for various stochastic demand patterns are shown in Table 5. In order to examine the behavior of the ESF strategy in relation to unit costs c_I and c_b, we have performed a cost analysis. Since the fitness function of the ESF strategy is a function of unit costs, the supervisory controller should be trained for every different value of unit costs. The results are shown in Table 6, where the ESF control approach is compared with the HSF and the EDF approaches for various values of c_I and c_b.

Figure 11 presents the fitness of the best two individuals (solid and dashed line) for this approach. The mutation rate is 0.1 while the population size is 40 as in all previous test cases presented in this work.

6.2.2 Test Case 4: Cost analysis for single-part-type production line

As we have seen in Section 3 the fitness function of the ESF strategy is a function of unit costs. This means that the supervisory controller should be trained for every different value of unit costs in order to optimize the systems performance. We have performed a number of experiments in order to examine the behavior of the ESF strategy in case of unit costs c_I and c_b variation. The values of the production line parameters are the same as in Section 6.2.1. We have examined the case of stochastic demand

Table 5 Comparative results for test case 3 for various values of stochastic demand (c_I=c_b=0.5)

Demand		HSF			ESF		
		$\overline{WIP}$	$\overline{BL}$	C	$\overline{WIP}$	$\overline{BL}$	C
Stochastic	0.5	19.886	0.167	10.027	6.446	0.32	3.383
	1	11.722	2.584	7.153	9.17	3.853	6.512

Table 6 Comparative results for test case 4 for various values of c_I, c_b (d=1)

c_I	c_b	HSF		ESF	
		C	$\overline{WIP}$	$\overline{BL}$	C
0.75	0.25	9.438	8.428	4.255	7.385
0.5	0.5	7.153	9.17	3.853	6.512
0.25	0.75	4.869	11.378	2.304	4.573

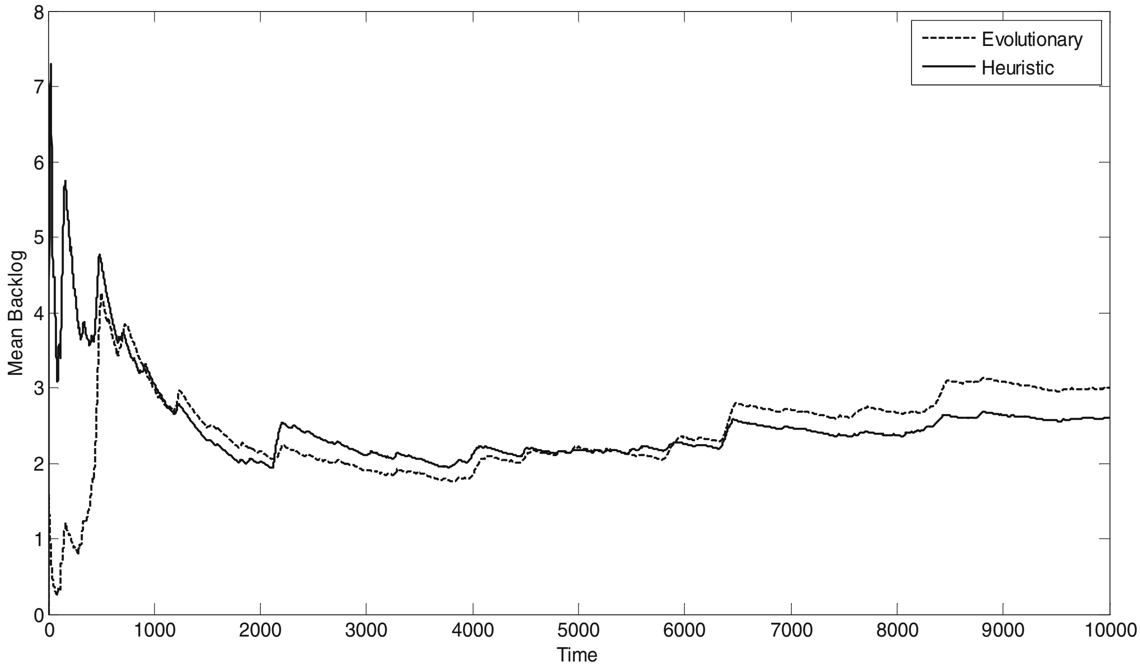


Fig. 10 Evolution of $\overline{BL}$ in the production line of test case 1 with stochastic demand (d=1)

with mean demand rate equal to one. The results are shown in Table 6, where the ESF control approach is compared with the HSF approach for various values of c_I and c_b.

6.2.3 Test case 5: single-part-type production networks

Finally, we have examined the performance of the supervised fuzzy evolutionary approach for the case of the production network presented in Fig. 7. The values of the production network parameters are the same as in Section 6.1.2.

Table 7 shows comparative results of $\overline{WIP}$, $\overline{BL}$ and C, when c_I and c_b are equal to 0.5, for various stochastic demand patterns. In this test case, we have also examined the sensitivity of evolutionary strategies in slight or moderate variations of demand. The comparative results of ESF and HSF strategies for $\overline{WIP}$, $\overline{BL}$ and C (c_I=c_b=0.5) in relation to demand are presented in Table 8. The ESF and EDF controllers used in this experiment are trained for stochastic demand equal to 2.

6.3 Olive oil production system: a real test case

In the final test case we examine the production system shown in Fig. 8. It represents a pomace oil processing plant located in Crete, Greece. The pomace oil unit receives the pomace in bulk quantities, usually tens of tones, from various olive mills. There are three stages in the production systems shown in Fig. 8: the *drying process*, the *oil extraction* process and the *neutralization*.

In the drying process, the pomace (which is a mixture of pomace oil (5%), water (50%) and pomace wood (45%)) is fed into rotating cylinders that are heated. There, the large content of water evaporates and is removed by fans in the form of wet steam. The removal of the water is of primal importance, since the oil cannot be extracted with water present. A detailed discussion on the control of the pomace drying system for this plant is presented in [14]. After drying, oil is solvent-extracted from the dried (7–8% water) pomace. This procedure is common for most seed oils. Hexane is used in order to remove the oil from pomace. The mixture of pomace oil and hexane is heated at low temperature so that the liquid hexane evaporates and the

Table 7 Comparative results for test case 5

Demand		HSF			ESF		
		$\overline{WIP}$	$\overline{BL}$	C	$\overline{WIP}$	$\overline{BL}$	C
Stochastic	0.5	5.43	0.09	2.76	1.406	1.888	1.647
	1	7.162	0.505	3.834	3.036	2.681	2.859
	2	14.555	2.777	8.666	8.914	5.55	7.232

Table 8 Sensitivity analysis for production network of test case 4 in relation to demand

Demand	HSF			ESF		
	$\overline{WIP}$	$\overline{BL}$	C	$\overline{WIP}$	$\overline{BL}$	C
1.0	6.962	0.514	3.738	28.593	0.105	14.346
1.5	10.426	1.348	5.887	25.492	0.471	12.98
1.7	12.081	1.785	6.933	7.932	3.733	5.833
1.9	13.748	2.607	8.178	8.618	5.318	6.968
2	14.321	3.059	8.69	9.192	6.047	7.62
2.1	15.597	3.497	9.547	9.876	6.958	8.417
2.3	17.39	4.544	10.967	11.384	9.103	10.244
2.5	17.728	6.852	12.29	14.639	7.566	11.102
3.0	20.01	28.75	24.38	33.06	19.575	26.316

crude pomace oil remains. What is left after the crude oil extraction is called pomace wood. Pomace wood is used as fossil fuel and has about one-third of the heating value of petrol. Neutralization is the next process. When pomace oil is neutralized, sodium hydroxide, also known as caustic soda, is added to lower the acidity. This neutralizes the bitter taste of the crude oil and creates a sodium salt which is then separated out from the oil and used for soap stock.

The production network shown in Fig. 8, produces three different end products, namely soap stock, neutralized olive pomace oil and pomace wood, and for presentation reasons, it is considered as a single product system. This is because all three end products are produced simultaneously. The demand that the system should satisfy is 0.16 t/h. The maximum production rates for soap stock, pomace oil and pomace wood are μ_1=10.9 t/h, μ_2=10 t/h, μ_3=25 t/h, respectively. The repair rates are all equal to rr_i=0.5 and the failure rates are p_i=0.05. The last two production stages (extraction and neutralization) are actually disassembly stages. The second machine processes dried olive stones and produces crude olive pomace oil and pomace wood. The disassembly factors are 0.1 and 0.9, respectively. This means that from 1,000 kg of raw material is produced 100 kg of crude olive pomace oil and 900 kg of pomace wood. In the third stage, the crude olive pomace oil is processed in order to produce soap with a disassembly factor 0.2 and neutralized olive pomace oil with disassembly factor 0.8. The buffer capacities are $BC_{1,2}$=50 t, $BC_{2,3}$=40 t, $BC_{F,1}$=60 t, $BC_{F,2}$=100 t, $BC_{F,3}$=100 t. Comparative results are presented in Table 9 for the distributed fuzzy and Table 10 for the supervised fuzzy approach.

6.4 Statistical significance of results

Ten simulation runs of 10,000 time units each have been carried out. Table 11 presents the maximum relative error of $\overline{WIP}$ estimates, with level of significance α=0.05, for the case of single-part-type production network.

7 Discussion

Based on the obtained results, the following observations are made:

The evolutionary approaches achieve a substantial reduction of WIP in almost every case. On the contrary, there is a small increase of backlog in most cases (see Tables 2, 4, 5 and 7).

These results are due to the fact that WIP and backlog are competitive measures of production system's performance. In order to reduce backlog, one has to increase the system's throughput and thus increase its WIP. When

Table 9 Comparative results for the distributed controlled olive pomace oil production system

Demand		HDF			EDF		
		$\overline{WIP}$	$\overline{BL}$	C	$\overline{WIP}$	$\overline{BL}$	C
Constant	0.16	377.89	0.00065	188.94	169.25	0.03	84.64

Table 10 Comparative results for the supervised controlled olive pomace oil production system

Demand		HSF			ESF		
		$\overline{WIP}$	$\overline{BL}$	C	$\overline{WIP}$	$\overline{BL}$	C
Constant	0.16	240.403	0.00065	120.2	27.464	0.1554	13.81

Table 11 Maximum relative errors of $\overline{WIP}$ estimates for single-part-type production network

	HDF	HSF	EDF	ESF
$\overline{WIP}$(%)	2.49	4.45	1.39	2.74

demand is very high, one may consider that the service rate and thus the backlog is more important than WIP. When demand can be easily satisfied and backlog is at low levels, a substantial reduction of WIP may be more important than a small increase in backlog. With the use of the evolutionary algorithm the system's performance becomes more balanced. In all cases the sum of $\overline{WIP}$ and $\overline{BL}$ is reduced.

This may be seen more clearly in the results of the production cost analysis. By observing Table 3, for test case 1, one can see that the heuristic approach may give better results only when the inventory unit cost c_I is much smaller than the backlog unit cost c_b.

As has already been mentioned, the heuristic approach may achieve lower costs in some cases. This is basically due to the choice of the fitness function of the evolutionary tuned system. The fitness function chosen in the case of EDF approach gives equal importance to $\overline{WIP}$ and $\overline{BL}$. When the contribution of WIP to the production cost is much greater than the contribution of backlog, the HDF approach may outperform the EDF approach.

In the ESF strategy, the fitness function is selected in such a way that the total production cost should be minimized. A careful examination of Tables 3 and 6 shows that this is not always true. The ESF scheme in general outperforms the HSF scheme, but in comparison to the EDF approach ESF performance is in general worse (see Tables 3, 5, 6 and 7). This is not surprising since the lower level distributed fuzzy controllers are not optimized in case of ESF control approach. As a result, the ESF approach is less time consuming than EDF but it is also less effective.

The sensitivity analysis presented in Table 8 shows that slight and moderate demand changes do not affect the evolutionary strategies performance dramatically. In general, the ESF performance is worse in comparison to HSF only when demand has a high deviation (more or about ±50%) from d=2, which is the value of demand ESF was trained to satisfy. This result is in accordance to intuition, since heuristically selected fuzzy controllers (HDF and HSF) are generally expected to work for a wider set of demands in comparison to the optimized (EDF and ESF) ones.

8 Conclusions

An evolutionary algorithm strategy for the optimization of already established fuzzy production control architectures [4, 5] has been presented. The EA strategy selects the membership functions of the fuzzy controllers in a way that WIP and backlog values minimize fitness function based on production surplus. Simulation results, for a number of taste cases, have shown an important improvement of performance and production related costs, with the use of EA strategies. More specifically, the EA strategies manage to reduce substantially the weighted sum of WIP and backlog and thus improving the inventory and backlog costs. Evolutionary algorithms clearly represent a successful approach towards the optimization of fuzzy production control approaches.

In the future, it would be very interesting to consider the case of seasonal demand. Another interesting extension would be the use of EA strategies in more complex production systems such as multiple-part-type and/or reentrant systems.

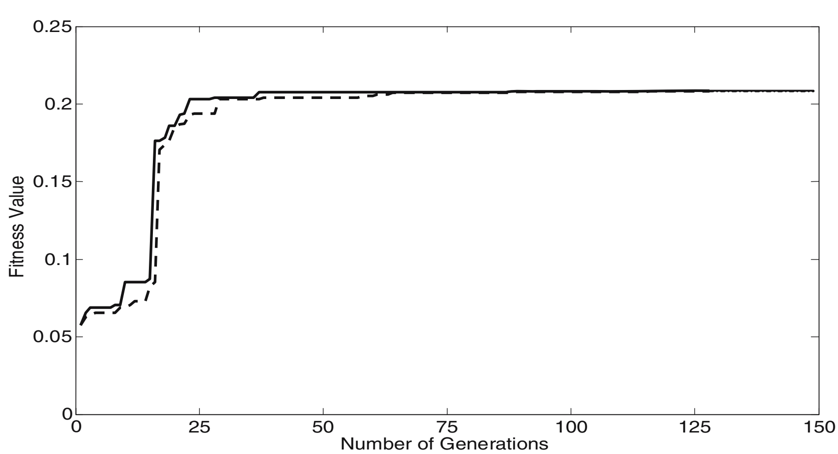


Fig. 11 Fitness functions of the two best individuals for single product line ESF approach

References

1. Conway R, Maxwell W, McClain JO, Joseph TL (1988) The role of work-in-process inventory control: single-part-systems. Oper Res 36:229–241
2. Bai SX, Gershwin SB (1994) Scheduling manufacturing systems with work-in-process inventory control: multiple-part-type systems. Int J Prod Res 32:365–386
3. Gershwin SB (1994) Manufacturing systems engineering. Prentice Hall, New Jersey
4. Tsourveloudis NC, Dretoulakis E, Ioannidis S (2000) Fuzzy work-in-process inventory control of unreliable manufacturing systems. Inf Sci 127:69–83
5. Ioannidis S, Tsourveloudis NC, Valavanis K (2004) Fuzzy supervisory control of manufacturing systems. IEEE Trans Robot Autom 20:379–389
6. Custodio L, Senteiro J, Bispo C (1994) Production planning and scheduling using a fuzzy decision system. IEEE Trans Robot Autom 10:160–168
7. Passino KM, Yurkovich S (1998) Fuzzy control. Addison-Wesley, Menlo Park, California
8. Klir GJ, Yuan B (1995) Fuzzy sets and fuzzy logic: theory and applications. Prentice Hall, New Jersey
9. Jain AK, Elmaraghy HA (1997) Production scheduling/rescheduling in flexible manufacturing. Int J Prod Res 35:281–309
10. Tedford JD, Lowe C (2003) Production scheduling using adaptable fuzzy logic with genetic algorithms. Int J Prod Res 41:2681–2697
11. Gordon O, Herrera F, Hoffmann F, Luis M (2001) Genetic fuzzy systems: evolutionary tuning and learning of fuzzy knowledge bases. World Scientific Publishing Co. Pte. Ltd, U.K
12. Homaifar A, McCormick E (1995) Simultaneous design of membership functions and rule sets for fuzzy controllers using genetic algorithms. IEEE Trans Fuzzy Syst 3:129–139
13. Kouikoglou VS, Phillis YA (1997) A continuous-flow model for production networks with finite buffers, unreliable machines, and multiple products. Int J Prod Res 35:381–397
14. Tsourveloudis NC, Kiralakis L (2005) Modeling and control of a rotary drying process. WSEAS Trans Syst 4:2361–2368