Adaptive Learning of Process Control and Profit Optimization Using a Classifier System

A. H. Gilbert*
School of Computing and Maths
University of Teesside
Middlesbrough
Cleveland, TS1 3BA
United Kingdom

Frances Bell†
School of Computing and Maths
University of Teesside
Middlesbrough
Cleveland, TS1 3BA
United Kingdom

Christine L. Valenzuela
School of Computing and Maths
University of Teesside
Middlesbrough
Cleveland, TS1 3BA
United Kingdom

Abstract

A classifier system is used to learn control and profit optimization of a batch chemical reaction. Ability to learn different market conditions and changes to reaction parameters is demonstrated. The profit sharing algorithm is used for apportionment of credit. The greater effectiveness of the use of the genetic algorithm over apportionment of credit alone or the random replacement of low strength rules is also shown. The classifier system is unusual in having more than one action per rule.

Keywords

Apportionment of credit, batch process, chemical process, classifier system, genetic algorithm, market, profit-sharing plan, return on capital.

1. Introduction

Classifier systems (CS) are a class of self-learning expert systems for which the production rules are coded so as to be processable by a genetic algorithm (GA) (Holland, 1962; Holland & Reitman, 1978). These rules/chromosomes interact with the environment, which is to be learned, in an apportionment of credit (AOC) subsystem resulting in the modification of rule strengths/fitnesses in accordance with their performance in the environment. Where more than one rule is appropriate to the current environment state, use of the rule of higher strength is more probable. Following a period of environmental interaction, a subset of the rules is replaced by new rules evolved by the GA. The probability of an old rule being a

* Send correspondence to A.H.Gilbert@tees.ac.uk.
† Current address: Dept. of Applied Science & Computing, University College Salford, Frederick Road, Salford, M6 6PU, U.K.

parent of a new rule is proportional to its fitness and hence it is expected that the population will improve by the innovative combination of the strongest genetic material.

The utility of CSs for control of complex environments has been demonstrated in a number of applications including space vessel power management (Goodloe & Graves, 1988), gas pipeline design (Goldberg, 1983), and docking of a tractor-trailer (Raed & Hassoun, 1993).

There are several studies of adaptive control of a batch chemical reactor (Katoh, Nakao, & Hanawa, 1989; Regev, Lewin, & Lavie, 1989; Rao & Lee, 1991) in which systems learn to maintain a predetermined optimum temperature profile. Ryhiner, Dunn, Heinzle, and Rohani (1992) have studied adaptive learning in a biochemical process by the method of steepest ascent. Chen and Weigand (1992) have used a neural net to simulate a batch biochemical reaction and a GA to optimize its reaction profile. This article presents a preliminary study of a CS for adaptive control of a chemical batch reaction with no predetermined profile; a complex, realistic profit optimization is the objective function.

2. The Environment to Be Learned

2.1 Choice of Application

Optimal control of batch processes is normally of great importance because they are typically used for manufacture of high-cost, low-volume materials. Batch processes are generally more difficult to control than a large-scale continuous process because there are no steady-state conditions. Even if all mechanisms and parameters of the reactions involved were known, it would be difficult to optimize profitability; in addition, in the real world, there are complications owing to unpredictable changes that may be equipment related (fouled heat-exchange surfaces, sticking valves), materials related (purity and cost), or market related (market size/share, product price) so that the batch may enter a state that was previously unknown:

> More than a dozen times a month over the past few years, the operators at a certain chemical plant encountered what were considered unusual problems in running their batch reactors. Records show that about 15% of the time a process expert was available to give advice; the other 85% of the time the operators were on their own. Records show that between the experts and the operators, solutions were wrong about three times per month—sometimes because of errors in judgement, in other instances because the actions were executed too late. The money lost due to these incidents amounts to more than $260,000 annually. (Nisenfield & Turk, 1986, p. 57)

Nisenfield and Turk then continue by advocating decision-support expert systems (ESs). While this would be an eminently valid, potential improvement, it would entail writing an ES for every process operated by the company. They might need frequent partial rewrites as, for instance, market conditions or raw materials purity change. Experts (and hence ESs) may not be able to anticipate all "unusual problems." A robust CS might be a better solution and to this end we explore the application of a CS to a batch process.

2.2 Dynamics, Profitability, and Simulation of a Batch Chemical Process

Chemical processes need to cope as efficiently as possible with undesirable reactions (decomposition, overreaction, wrong reaction) that inevitably occur in parallel with the desired

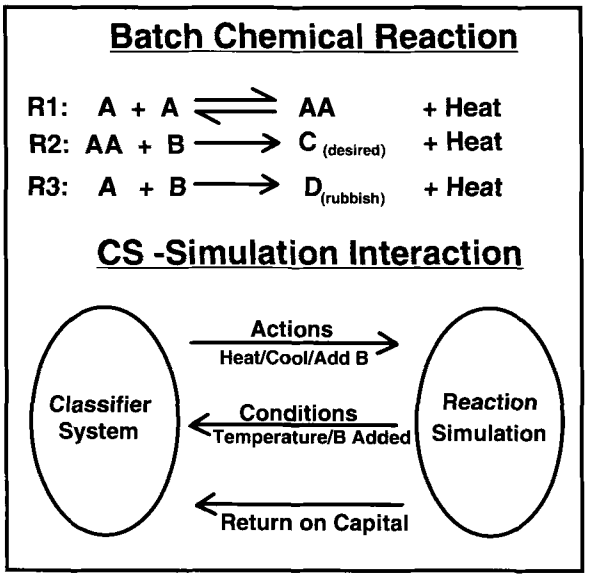


Figure 1. *Chemical reaction system and classifier system interaction.*

reaction(s). In cases where a sequence of reactions is necessary to obtain the desired product and where an intermediate product is difficult or costly to isolate, it is often advantageous or necessary to carry out the sequence in the same batch process. This may be even more beneficial where a reaction early in the sequence (with a product difficult to isolate) is a reversible reaction, because the further reaction of this intermediate product prevents it from taking part in the reverse reaction.

The hypothetical process chosen for study (Figure 1) involves some typical features described above. It consists of two sequential reactions that give a desirable product and a parallel reaction that gives an unwanted product. The desired product, C, can only be formed by reaction (R_2) of reactant B with an intermediate product AA formed by dimerization (R_1) of reactant A. Undesirable product, D, will be formed if B reacts (R_3) directly with A (in a real case there would typically be several unwanted parallel reactions). There is a temperature threshold below which R_1 will not proceed, hence for efficiency, the CS must learn to raise the temperature before significant addition of B takes place. Also, R_1 is a reversible reaction in which AA can decompose back to A; this reverse reaction is favored by high temperature. All three reactions liberate heat, making temperature control important. The CS must also learn that the process must not be allowed to exceed a temperature of 425°K.

The process envisaged would be a small scale pharmaceutical or specialty chemical manufactured in equipment as illustrated in Figure 2.[1]

Typically such processes are optimized by statistical replicate experiments or "evolutionary operation" (experimental steepest ascent around the best known point). The model calculates temperature changes (caused by controls, heat loss, and reaction), chemical concentrations (using reaction-rate equations), and profit.

Changes in external heating (and cooling), for control purposes, are calculated by multiplying the total available heat (or cooling) by the fractional opening of the appropriate valve

1 The heating/cooling system has been drawn to reflect the classifier action bit structure. It would not be built this way.

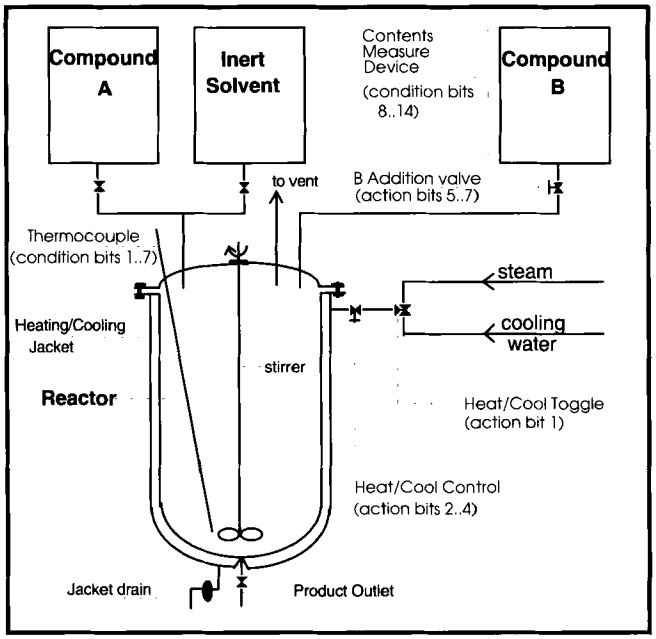


Figure 2. Plant diagram showing condition sensors and action controls.

and by the temperature differential. Heat loss from the reactor is calculated similarly

$$\frac{\delta T}{\delta t} = \frac{H_m \frac{V_h}{8}(393 - T) + W_m \frac{V_c}{8}(T - 303) - 0.04(T - 298)}{MS}$$

where H_m and W_m are the maximum heating and cooling available (for simplicity, parameters such as heat-transfer coefficients and surface area are rolled up into these values), V_h and V_c are the positions of the heating and cooling valves (see below), 393°K, 303°K, 298°K are the steam, cooling water, and ambient temperatures, M is the mass of total reactor contents and S the specific heat.

Reaction rate equations are of the form

$$\frac{\delta C_z}{\delta t} = A_0 e^{-E/RT} C_x C_y \qquad \text{for reaction } (R_n)\ X + Y \rightarrow Z$$

where C_X is the concentration of component X, A_0 is a constant, E is the activation energy of the reaction, R is the gas constant, and T is temperature (°K) (values used are in Table 1).

The temperature rise owing to heat liberated in each reaction is determined by equations such as

$$\frac{\delta T}{\delta t} = \frac{-H_r}{MS} \frac{\delta C_z}{\delta t}$$

where H_r is the heat of reaction[2] (H_r values are in Table 1).

The major simplifications made in the reaction simulation are that all reactants are treated as nonvolatile, the specific heat (which would with vary with concentration and

2 By chemical convention, heats of reaction where heat is liberated are negative.

Table 1. Reaction constants used in simulator.

	Scenario	$R_1^f : A + A \rightarrow AA$	$R_1^r : AA \rightarrow A + A$	$R_2 : AA + B \rightarrow C$	$R_3 : A + B \rightarrow D$
$A_0(1\ mole^{-1}\ sec^{-1})$	1, 2, 3	3×10^{10}	9×10^{10}	1.3×10^{11}	1.2×10^{11}
$E(kJ\ mole^{-1})$	1, 2	87	100	83	87
	3	"	"	83	84
$H_r(kJ\ mole^{-1})$[4]	1, 2	−193	193	−255	−235
	3	"	"	−750	−600

temperature) is constant, and heat transfer area (which would change with addition of B) is constant.

The profitability is calculated on a yearly basis assuming that the achieved result at the time of calculation is employed all year. The plant may not make more than the market size. Variable cost is calculated from materials and energy used on a similar yearly basis. Fixed costs are maintenance and depreciation. Labor costs are not included because the CS replaces the human operator!

Profit is expressed as return on the capital value of plant used (cost parameters are specified in Table 2).

$$\text{Sales Income } I = U_C P_C \frac{t_y}{t_b} \qquad U_C \frac{t_y}{t_b} \leq L$$

$$\text{Variable Cost } V = (U_A P_A + U_B P_B + U_D P_D + U_S P_S + U_W P_W)\left(\frac{t_y}{t_b}\right)$$

$$\text{Fixed Cost } F = Q + Cap\,\frac{d}{100}$$

$$\text{Return (\%) } R_C = 100\frac{(I - V - F)}{Cap}$$

where U_A, U_B, U_C, U_D are the mass of A, B, C, D used or made in the batch; P_A, P_B, P_C are the unit price of A, B, C; P_D is the cost of disposing of D; U_S, U_W are steam and water used; and P_S, P_W their unit price. L is the size of the market for C, Cap and Q are the capital cost and annual maintenance cost of the equipment, d is the percent annual depreciation charge and t_b and t_y are the time for a batch and the length of a year, both in minutes.

The changes in reactor contents, temperature, and return on capital during simulation of a batch reaction, (illustrated in Figure 3) were produced during a test run of the simulation under one particular regime of valve settings. No change in chemical concentration (Figure 3a) occurs for approximately 10 minutes until the temperature (Figure 3b) reaches 325°K (threshold for reaction R_1, see above). At this point reactant A rapidly diminishes as it dimerizes to form the intermediate product AA, which, typically of an intermediate, rises and then disappears owing to further reaction with reactant B. Desired product C and unwanted product D are seen to increase up to the point (approximately 60 minutes) where the concentration of A and AA are so low that the reaction rate is almost imperceptible. The return on capital graph (Figure 3c) shows what the return would be if the batch had been stopped at the indicated time and all batches for a year run in exactly the same way. (In this

Table 2. Cost parameters.

Parameter	Symbol	Value
Market Size(kg/yr) Scen 1	L	Unlimited
" " Scen 2,3	L	40000
A Price(£/ Kg)	P_A	2
B Price(£/ Kg)	P_B	6
C Price(£/ Kg)	P_C	8
D Disposal Cost(£/Kg)	P_D	0.5
Heat Cost(£/kJ)	P_S	2×10^{-4}
Cool Cost(£/kJ)	P_W	4×10^{-5}
Maintenance(£)	Q	20000
Capital(£)	Cap	400000
Depreciation(%)	d	10
InterbatchDeadTime(min)		20

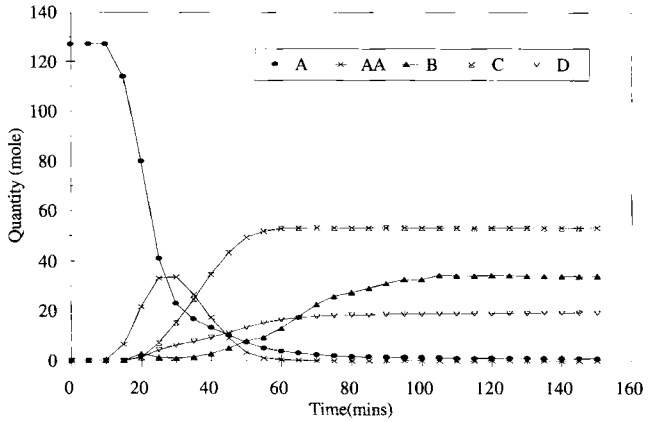
test illustration the simulation has been allowed to run on beyond the point where it should have been stopped (approximately 50 minutes) for best return [Figure 3c]).

2.3 Interaction of the Environment and the CS

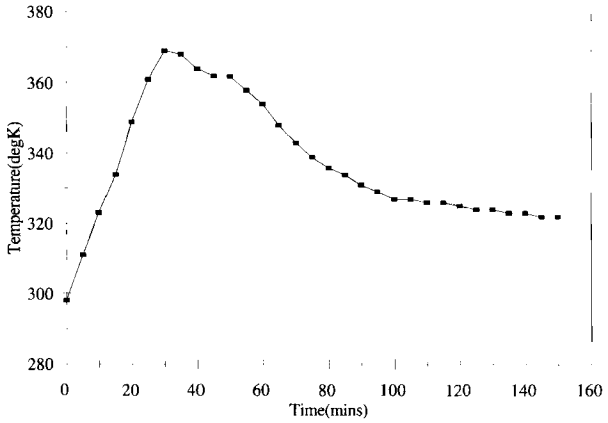
The environment-CS interaction is depicted in Figures 1 and 2. Interactions are of three types:

1. *Conditions*, passed from environment to CS, have been limited to the very easily measurable parameters temperature and quantity of component B added to the batch. These make up the condition part of the CS rules and are passed to the CS each five minutes of simulated time.
2. *Actions*, determined by the CS from the conditions, are control valve settings for steam heating, water cooling, and the addition of B. They are passed back to the environment, which will then operate under these conditions for the next five minutes.
3. *Return on capital* is the objective function, the basis on which the CS adjusts the strengths of all rules used in a batch in accordance with their performance.

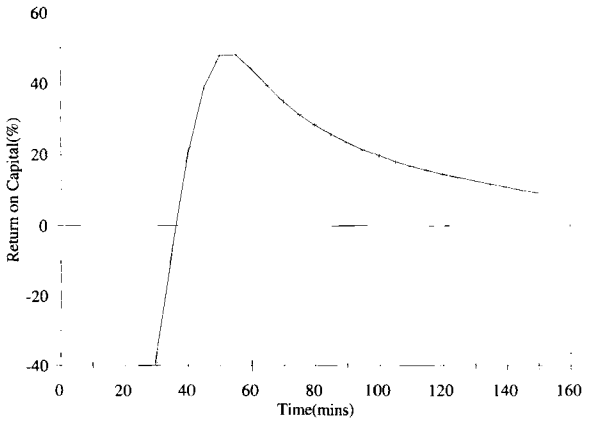
One area of interface control yet to be developed is that the CS should decide when to end the batch. A stop-action bit could easily be included but this has potential problems in that the random creation of classifiers would give approximately half with a stop bit set. More significantly the transfer of this bit during crossover to rules that had evolved for early reaction conditions could be very disruptive. In order to study the ability to learn control and optimization without this complication the artificial expedient of stopping the batch from the simulation has been adopted; currently the simulation stops itself, either after 125 minutes (25 control periods) or if an internal profit calculation shows a drop in profit from the previous period. Clearly this is unsatisfactory; a basis for work in progress in this area is discussed in Section 4.3.2.



(a)



(b)



(c)

Figure 3. Batch reaction simulation. (a) Chemical composition. (b) Temperature. (c) Return on capital.

Temp 298–425°K	B Added 0–127mole	Toggle Heat/Cool	H/C valve	Add valve
$c_1c_2c_3c_4c_5c_6c_7$	$c_8c_9c_{10}c_{11}c_{12}c_{13}c_{14}$	a_1	$a_2a_3a_4$	$a_5a_6a_7$
$c_n = \{0, 1, \#)\}$ (# = wild card)			$a_n = \{0, 1\}$	

Figure 4. Rule representation, relation between CS conditions and plant information, and CS actions and plant control.

3. The Classifier System

Much of the implementation of the CS is based on Goldberg's (1989) "Simple Classifier System" (SCS) but substantial changes have been made to accommodate profit sharing rather than bucket brigade in the apportionment of credit (see below), and the use of multiple actions in the classifiers, as well as considerable changes necessitated by factors related to the environment to be learned.

3.1 Rule Representation and Creation

Note that there is not a simple dependency of a single action on a single condition; for instance, the optimum heat/cool setting may depend on how much B has been added so far, as well as on the more obvious current temperature condition. The chosen rule representation is

Temperature condition, B added condition
: Heat/cool toggle action, Heat/cool valve setting action, B addition valve setting action

The coding of the rules is depicted in Figure 4. Each rule is coded as 21 bits. There are 14 condition bits, 7 (c_1–c_7) represent temperature (298–425°K) and 7 (c_8–c_{14}) the quantity of compound B added (0–127 mole).[3] There are 7 action bits. Because it would be inefficient to heat and cool at the same time, the first (a_1) is a toggle between heating and cooling,[4] a_2–a_4 represent the degree of opening of the steam (or water depending on the setting of a_1) valve and a_5–a_7 the B addition valve. In effect these valves are modeled as having eight operating positions from 0 (shut) to 7 (fully open). To obtain shorter effective schemata the conditions are coded with the most significant bit (MSB) on the right and the actions with the MSB on the left.

The initial population of 100 rules/chromosomes is created randomly. Gray's code is used for the conditions and actions; other workers (Hollstien, 1971; Frey & Slate, 1991) have reported indications of benefit over normal binary coding. The inclusion of "wild card" bits in the conditions enables the building of generalized rules. The number of "wild card" bits is set with probability of 0.5 (pwild, Table 3); this value, suggested by early experiments, gives 0 to 8 (usually 2 to 4) rules bidding at each interaction with the simulation.

3 A "mole" is the molecular weight of a compound in grams.
4 Purists might argue that, by this coding, the CS is being told an important rule, "do not cool and heat at the same time," which it should learn for itself.

Table 3. Classifier system parameters.

Parameter	Value
PopulationSize	100
Initial Strength	10
ReplaceSize	20
PMutate	0.02
PWild	0.5
BidSigma	0.12
BidTax	0.0002
LifeTax	0.005
Punish	−2
Gradient	0.3
Const	3.3
BetterBonus	8
EqualsBonus	4
WorseBonus	−1

3.2 Apportionment of Credit

Each rule, for which the conditions match the message from the simulation, bids for its actions to be used for control in the next period; the deductible bid is risked in the hope of gain by a reward for making a successful contribution. Bids are proportional to the fitness of the bidders but are reduced in proportion to the rule's generality and are perturbed by noise (BidSigma, Table 3), and the highest bidder is selected (for detail see Goldberg, 1989).

If no match is found a new rule is created with conditions that match the message but with some bits randomly replaced by "wild cards"; its actions are random. It is given the average strength of the current population, used for the current control period and added to the population. If more than 75 new rules are created in the course of a generation, that generation is abandoned (so far this has not occurred).

In the traditional expert system the relative ratings of the rules are fixed by the expert. In the CS the relative rule strengths must be learned. The system must, however, discover and learn innovative combinations of rules; hence a reward of combinations rather than individuals is required. CSs have mainly used one of two schemes to distribute reward. The "bucket brigade" (BB) algorithm (Booker, 1982) has gained favor of late but an earlier algorithm (Holland & Reitman, 1978) developed and named the "profit-sharing plan" (PSP) by Grefenstette (1988) has been used in this work. Grefenstette's comparative study concluded that the BB is better when rules fire in parallel and the PSP is better when there is a single, active chain. In addition, there are problems with the BB where long chains are involved (Holland, 1985). The single, active, long chain of the batch reaction would appear to make PSP a better choice here.

The combination of rules used in a batch are first rewarded in proportion to the size of return in profit they produce; in addition there is a bonus reward, the bonus level depending on bettering, equaling, or doing worse than the best result so far achieved in that generation. The reward and bonus are shared by the combination of rules, with more specific rules favored by a linear scaling. (Although Grefenstette's (1988) original formulation of PSP

gave each rule the full reward, it seemed reasonable in our case that the rule combination should share the reward, so favoring shorter chains.) Rules used more than once get a multiple reward. If the batch has been stopped by the simulator owing to a fall in the profit compared with the previous period (see above) the last rule is not rewarded. If a rule-set exceeds the permissible temperature (425°K) a punishment (−2) is shared. In addition, at the end of each generation a small life tax is extracted from all rules so that rules that do not bid lose strength. For clarification, the pseudocode (see Table 3 for constants) is

```
IF  MAXTEMP exceeded
       reward := PUNISH
       bonus   := WORSEBONUS
ELSE
       reward:= GRADIENT*ThisReturnOnCap + CONST
       IF reward < 0 THEN reward:=0
ENDELSE
IF ThisReturnOnCap >  BestThisGen
       bonus:= BETTERBONUS
ELSE IF ThisReturnOnCap = BestThisGen
       bonus:= EQUALSBONUS
ELSE
       bonus:= WORSEBONUS
reward:= reward + bonus
```

The payment of this reward changes the strength of each contributing classifier by

```
strength := strength +
            (reward / nClassifiersInSet) * (0.8 + 0.2
            * nNonWildCards / nConditionBits)
```

3.3 The Next Generation

The subset of rules/chromosomes to be changed each generation has been 20 in work so far. Selection of the parents is by a roulette-wheel technique biased in proportion to the rule strengths (Goldberg, 1989). Each pair is subjected to single-point crossover with random choice of crossover point. Mutation of random bits in the offspring also occurs, set by `pmutate` at a low probability, 0.02. After offspring are born they are checked against the other offspring and the existing generation and are rejected and re-bred if they are a duplicate. They are each given the average strength of the two parents.

To keep the population at a steady size it is necessary to remove 20 members of the existing population plus rules equal to the extra number created in the AOC phase, where there was no match with the environment message. The choice of rules to die follows an adaptation of the Crowding Model (De Jong, 1975). First, all rules with strength less than 0.001 are removed (natural death). Then a subpopulation of potential rules to die, at least 20% larger than the number of rules that must die, is picked from the population. This subpopulation consists of all rules that have never bid in the AOC phase plus others chosen by a biased coin toss from rules below average strength (the bias is increased according to the number that must die). The subpopulation is then compared with the offspring population, matching bit for bit, and the most similar are removed. Because rules that have never bid are likely to be undesirable and unlikely to resemble the offspring subpopulation, there is a bias factor that increases the likelihood that a rule that has never bid will die.

4. Results and Discussion

Results typical of many runs are given in Figures 5, 6, and 7, which show the best result and the average result (of the last 20 batches of each generation) versus generation. The progress achieved in any run is to some extent dependent on the genetic material present in the initial, random population, but in all runs carried out there has been distinct evidence of learning. Profit normally rises steeply for the first few generations (because there is large scope for improvement) but then steadies out to a gradual rise with occasional setbacks. These setbacks are further discussed in Section 4.3.2.

4.1 Learning Different Environments

An essential feature of a CS is that it should not need modification in order to cope with changes in the environment. Ability to cope with changes in the market environment, market size, and price would be essential; it is also common practice that small-scale batch reaction equipment will be used for more than one product. Three scenarios are considered in the current work using the parameters in Tables 1 and 2.

Scenario 1: The Market Size Is Unlimited This scenario has been examined using 75 generations with 200 batches per generation and 200 generations with 75 batches per generation (Figure 5a,b). With an unrestricted market the CS tends toward the strategy of making compound C as fast as allowed by the restraints of the chemical dynamics and the compound B addition valve, achieving a best value of 52% return on capital in 45 minutes reaction time by raising the temperature rapidly and then adding compound B rapidly. The rule set that achieved this is given in Table 4 (note that rule 33 is used twice) and it is fairly clear that this is a highly generalized default set (recall that in the conditions the MSB is on the right). Some appreciation of the way rules have formed at generation 75 (200 batches per generation) can be seen in Table 5, which shows the stronger classifiers existing at that time. An indication that these strong rules are consistent with the above good strategy ("make C rapidly") can be seen by comparing the last two columns. The designations F, E, L refer to the first, early, and late periods of the reaction. There is a large measure of agreement between the reaction period determined by the conditions and the actions suited to these periods for the "make C rapidly" strategy. Note, however, that some rules (14, 24, 30, 33, 70, 74) are able to make mistakes (see Section 4.3.1).

The best rule set at generation 200 for the case shown in Figure 5b, where results have leveled out at 46%, is given in Table 6. This rule set was used for 71 of the 75 batches made in the generation. There are, however, other strong rules in existence and these are given in Table 7.

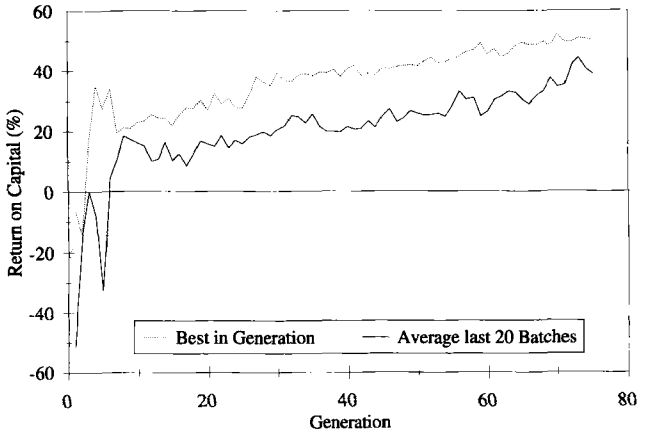
Probably the best action strategy for this scenario is

Heat (H) 7; H 7; H 7; H 7 Add (A) 7; H 7 A 7; H 5 A 7; H 5 A 7; H 5 A 7; H 5 A 4

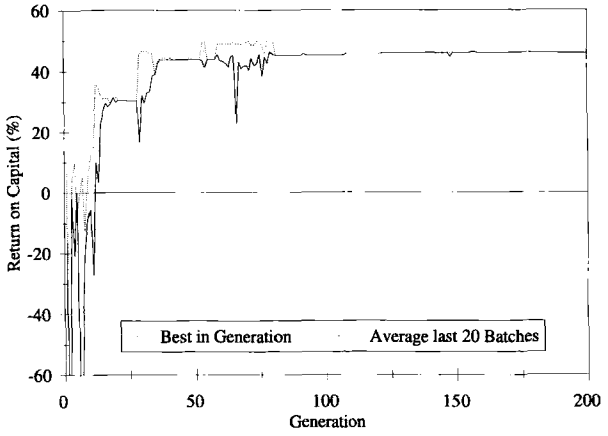
giving a return of 55.6% by making 79,114 kg/year of C in 6,492 batches.

Scenario 2: The Market Size Is Limited to 40,000 kg/year In this case no extra profit may be generated as a result of potential to make more than the market limit so, clearly, less total profit can be made.[5] Less importance must, therefore, be attached to speed and more

5 In the real world, if, as discussed above, the equipment can be used for other products, there might still be an advantage in fast production.



(a)



(b)

Figure 5. Scenario 1, market unlimited. (a) 200 batches per generation. (b) 75 batches per generation.

Table 4. A rule set for Scenario 1, best rule set found.

Rule#	Strength	Conditions		Actions	Decoded Actions
33	4.405	******0	*****00	1 100 001	Heat 7 Add 1
33	4.405	******0	*****00	1 100 001	Heat 7 Add 1
70	6.430	*0****0	0****00	1 100 010	Heat 7 Add 3
9	5.701	1*****0	*****00	1 100 010	Heat 7 Add 3
43	5.280	******0	*****00	1 100 101	Heat 7 Add 6
65	4.942	****01*	******0	1 110 101	Heat 4 Add 6
64	7.734	****011	*0**110	1 010 101	Heat 3 Add 6
25	5.277	*****1*	*****10	1 011 100	Heat 2 Add 7
36	1.582	*******	******0	1 011 101	Heat 2 Add 6

Table 5. Strong rules formed after 75 generations (Figure 5a).

Rule#	Strength	Conditions	Actions	Condition Period	Action Period
2	5.489	**1**** 0******	1 100 010	E L	E
10	5.117	*****1* *****10	1 011 100	L	L
8	4.741	*****11 011**1*	1 000 101	L	L
14	4.911	******0 *****00	1 100 101	F E L	L
23	8.343	****011 *0**110	1 010 101	L	L
24	5.321	*****01 ******0	1 110 101	E L	L
25	6.453	*0****0 0****00	1 100 010	F E	E
33	5.796	***0**0 *****00	1 100 101	F E L	L
36	4.020	******* 101*011	1 011 011	L	L
45	7.302	***1*11 *1***10	1 000 100	L	L
48	7.839	***1*11 ****010	1 100 100	L	L
50	7.214	*01***0 ****100	1 100 001	L	E L
51	7.525	***1*11 ***001*	1 000 101	L	L
66	6.628	***0**0 0****00	1 100 010	F E	F E
62	4.887	******1 ***001*	1 100 010	L	L
70	5.705	1*****0 *****00	1 110 010	E L	E
76	6.843	**0**00 *****00	1 100 101	F E L	L
86	6.997	101***0 *****00	1 100 011	E L	E
87	5.532	***1**0 *****00	1 100 100	E L	L
88	6.010	***1*11 **0*010	1 100 110	L	L
92	6.606	**1***0 ****100	1 100 101	L	L
95	4.842	1****11 ****010	1 100 100	L	L
97	6.299	*****11 ***001*	1 000 101	L	L
98	5.633	***1**1 1**001*	1 001 100	L	L

Table 6. Best rule set after 200 generations (Figure 5b).

Rule#	Strength	Conditions	Actions	Decoded Actions
1	12.185	0000*0* *0****0	1 001 111	Heat 1 Add 5
5	18.642	1*00*** 1011000	1 100 000	Heat 7
8	21.902	000*00* 1011*00	1 100 000	Heat 7
2	18.624	0*1**0* 1011000	1 100 000	Heat 7
3	21.902	1*1**10 1011*00	1 101 100	Heat 6 Add 7
6	18.642	0*11**1 **1*110	1 101 100	Heat 6 Add 7
4	13.893	0*0*011 1***010	0 001 100	Cool 1 Add 7
7	16.001	0*0*011 1***010	1 101 100	Heat 6 Add 7
13	13.893	1*1**11 1***010	1 100 100	Heat 7 Add 7

Table 7. Other strong rules after 200 generations (Figure 5b).

Rule#	Strength	Condition		Actions	Decoded Actions
56	6.351	0*1**0*	10****0	1 001 011	Heat 1 Add 2
57	7.264	0000*0*	*011000	1 100 000	Heat 7
58	6.925	000*00*	10****0	1 101 100	Heat 6 Add 7
59	6.925	0*0*011	1011*00	1 100 000	Heat 7
60	6.735	0*1**0*	1011*00	1 101 000	Heat 6
61	6.735	000*11*	1011000	1 100 000	Heat 7
63	6.736	1*1**10	1011*00	1 101 111	Heat 6 Add 5
68	7.574	0*0**0*	1011*00	1 100 000	Heat 7
69	7.574	000*01*	1*11000	1 001 100	Heat 1 Add 7
81	7.254	000*011	10**010	1 101 100	Heat 6 Add 7
82	7.254	0*0*011	1*****0	1 101 100	Heat 6 Add 7
87	6.527	000*11*	1011*00	1 101 111	Heat 6 Add 5
88	5.143	0000*0*	*0****0	1 001 100	Heat 1 Add 7
89	12.291	*00*001	**1*110	0 001 100	Cool 1 Add 7
90	12.291	0*11***	1011*00	1 100 000	Heat 7
91	9.541	0*1***1	1***010	1 100 100	Heat 7 Add 7
92	9.541	1*1**11	**1*110	0 001 100	Cool 1 Add 7
93	6.936	0000*0*	*0****0	1 001 000	Heat 1
94	7.551	0*1**0*	1011*00	1 101 111	Heat 6 Add 5
99	10.889	1*1**10	1011**0	1 101 100	Heat 6 Add 7
102	5.596	1*00*00	0*00*00	0 010 111	Cool 3 Add 5

to chemical efficiency, that is, make more C and/or less D and be parsimonious in the use of expensive compound B. In these circumstances the CS learns to take much longer over the reaction (90–120 minutes) and to add B more slowly. A major change in strategy takes place in that the CS learns to incorporate a significant period of "000" actions (do nothing) at the end of the reaction time, waiting for as much as possible of compound B, which has been added to enter into reaction. By these means the classifier regularly achieves around 25% return whereas, if the Scenario 1 strategy had been used, a return of only 20% would result. Results are given in Figure 6.

A best rule set discovered (shown in Table 8) makes the requisite 40,000 kg/year of C in 3,000 batches and achieves 25.9% return on capital.

Scenario 3: Changed Reaction Parameters and Market Size 40,000 kg/year As stated above, equipment of the type envisaged tends to be used for more than one product. In principle, it would appear that this CS could control any reaction that requires the controlled addition of a single component and that could be maintained within a desired temperature range by the particular constraints of available heating and cooling. Rather than build a new simulation to test this, it was expedient merely to change some of the reaction parameters (in reality these parameters are constants for a given reaction). In particular, the parameters made

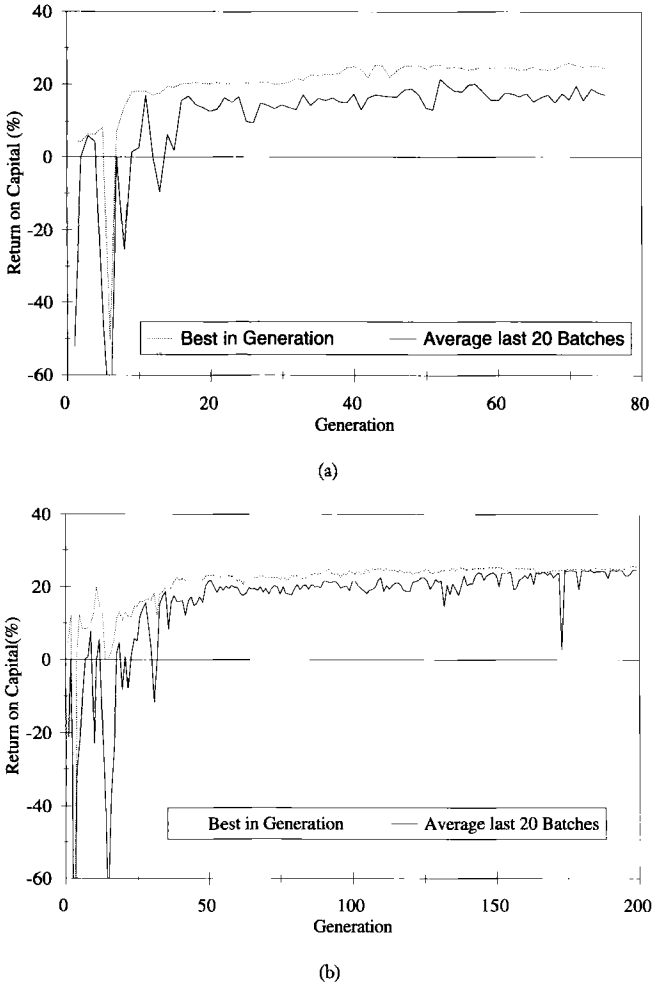


Figure 6. Scenario 2, market 40,000 kg/year. (a) 200 batches per generation. (b) 75 batches per generation.

a good return more difficult by making the production of D more favored as temperature increases (change in E, Table 1) and by increasing the heat liberated in reactions R_2 and R_3 (changes in H_r, Table 1). These latter changes also caused the system to exceed the upper temperature limit (425°K) quite frequently before it learned not to do so. From the rule set of Table 9, it can be seen that the CS responded by making significantly more use of cooling. In particular the system appears to have discovered that each time a significant rate of B addition is made, cooling must be applied to compensate for the heat liberated by reaction of this B because of the increased heats of reaction (this is precisely what a chemist would do). Some of the "do nothing" periods here are curious and probably parasitic because they are without cost: the first is clearly a waste of time; the consecutive group after the initial heating up of the contents may be a very subtle way of trimming a slightly hot reactor at no cost by heat loss but is more probably parasitic; the final ones are beneficial. The employment of divided reward in the PSP, to favor shorter rule sets, does not appear to have worked in

Table 8. Best rule set for Scenario 2.

Rule#	Strength	Conditions		Actions	Decoded Actions
45	2.548	00*****	*0*0*00	1 100 000	Heat 7
38	2.549	*****00	*0*0*0*	1 100 000	Heat 7
38	2.549	*****00	*0*0*0*	1 100 000	Heat 7
100	1.882	*****1*	*0*0*00	1 100 001	Heat 7 Add 1
84	1.941	*1**01*	*****00	0 000 000	Do nothing
31	2.076	*****1*	1*****0	0 000 011	Add 2
9	1.991	*****1*	*0****0	1 010 110	Heat 3 Add 2
52	2.226	*****1*	*1****0	1 000 011	Add 2
85	1.954	*****1*	1*****0	0 001 011	Cool 1 Add 2
18	2.089	****01*	*1****0	0 000 100	Add 7
13	2.214	****01*	******0	0 011 101	Cool 2 Add 6
54	1.610	*****1*	*0****0	1 000 100	Add 7
37	2.466	*****11	*1**01*	1 000 000	Do nothing
16	3.046	*****11	*1*001*	1 100 000	Heat 7
34	2.003	**1**1*	*1***1*	1 000 000	Do nothing
21	2.003	**1**1*	******0	0 011 101	Cool 2 Add 6
39	2.169	*****1*	***001*	1 000 000	Do nothing
17	3.256	**1**11	*01*01*	1 000 000	Do nothing
22	3.107	***0*11	*01*01*	1 100 000	Heat 7
40	2.746	*****11	*01*01*	1 000 000	Do nothing
40	2.746	*****11	*01*01*	1 000 000	Do nothing
43	0.899	**0**1*	****01*	1 000 000	Do nothing
89	1.256	*****1*	****01*	1 000 000	Do nothing
25	1.868	*****1*	*0**01*	1 000 000	Do nothing
25	1.868	*****1*	*0**01*	1 000 000	Do nothing

this case. Results are given in Figure 7 and the lower levels of return (approximately 20%) reflect the greater difficulty of this environment.

Ability to learn each environment from scratch is indicative of robustness. It is also desirable that the CS should be able to accommodate a change in the environment. Training was carried out as in Figure 5b (same random number seed) for 120 generations and at this point the market size was changed to 40,000 kg/year. The return dropped immediately to approximately 18% (Figure 8), as expected for the old rules applied to the new market size. Results then steadily improved to approximately 25% as the CS learned this new environment.

4.2 The Effect of the Genetic Algorithm

Figure 9 is typical of results obtained by running the AOC for 10,000 batches without benefit of the GA under Scenario 2. It can be seen that, although individual batches may achieve a moderate return at times, such results are not maintained. The results may be indicative of

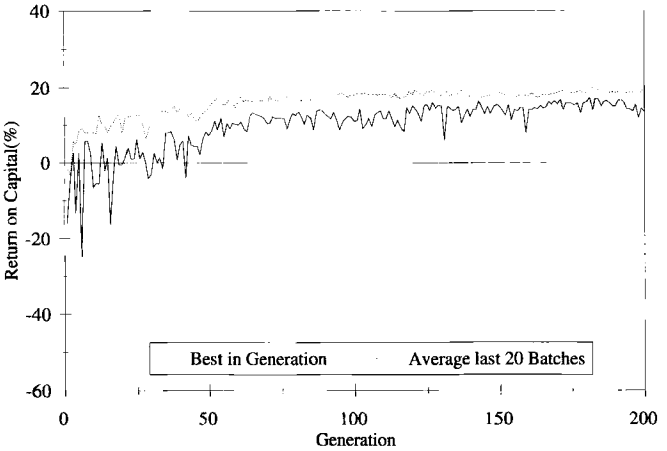


Figure 7. Scenario 3, high reaction heats, 75 batches per generation.

Table 9. Rule set for Scenario 3.

Rule#	Strength	Conditions		Actions	Decoded Actions
27	3.468	**0**00	***0000	1 000 000	Do nothing
5	4.417	**0**00	*0*0000	1 100 000	Heat 7
1	3.510	**0**00	***0000	1 100 000	Heat 7
8	3.629	**101**	***0000	1 100 000	Heat 7
21	3.762	**10*10	***0000	1 100 000	Heat 7
62	3.064	****010	***0000	1 000 000	Do nothing
20	3.776	*****10	***0000	0 000 000	Do nothing
20	3.776	*****10	***0000	0 000 000	Do nothing
98	3.405	*****10	***0000	1 000 000	Do nothing
2	3.771	***1*10	***0000	1 000 001	Add 1
35	2.251	***0*1*	******0	0 110 101	Cool 4 Add 6
35	2.251	***0*1*	******0	0 110 101	Cool 4 Add 6
65	2.467	*0***11	******0	0 101 100	Cool 6 Add 7
29	3.163	***1*11	****1*0	0 111 110	Cool 5 Add 4
29	3.163	***1*11	****1*0	0 111 110	Cool 5 Add 4
85	2.931	****011	****1*0	0 111 110	Cool 5
23	2.657	*0***11	******0	0 111 100	Cool 5 Add 7
14	2.493	*****11	******0	0 111 100	Cool 5 Add 7
76	1.692	*******	01*****	1 101 111	Heat 6 Add 5
9	4.017	*****11	1101**1	1 000 000	Do nothing
9	4.017	*****11	1101**1	1 000 000	Do nothing

a problem with generalists (see Section 4.3.1) in that one might expect a long AOC learning phase to reinforce more consistently and select the better results and hence increase the moving average.

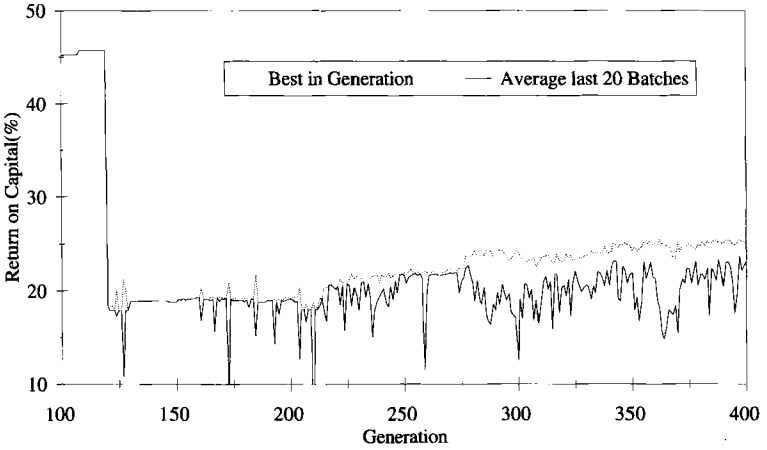


Figure 8. Change of environment, market size change at generation 120.

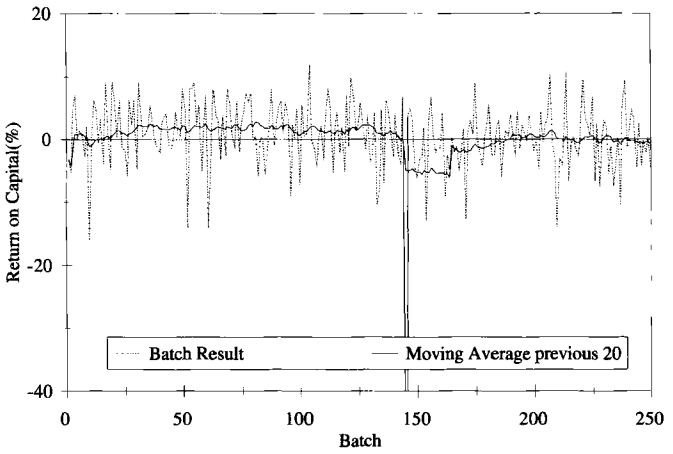


Figure 9. Operation of AOC without GA, Scenario 2.

Figure 10 shows the greater effectiveness of the GA over partial replacement by random rules. A run in which randomly selected, below average fitness rules were replaced by new random rules that were given average population fitness fluctuates wildly. (The same random number seed was used as in Figure 6b.) While occasional good results are obtained by this run, not using the GA, there is no sustained learning.

Overuse of the GA is also detrimental. Very short AOC learning (20 batches per generation) for 1,000 generations, although better than very long AOC with no GA, does not maintain the learning over many generations (Figure 11). It appears that a suitable compromise of AOC length is required to achieve effective selection, by allowing sufficient credit buildup but avoiding excessive polarization in the rule/ chromosome strengths (i.e., maintaining some gene-pool diversity). In the less extreme cases of 75 and 200 batches per generation, taking all runs for all three Scenarios, the shorter AOC appears more efficient, in that it tends to achieve a respectable return after fewer batches (2,500–3,000) than the 200 batch per generation case (3,000–5,000 batches).

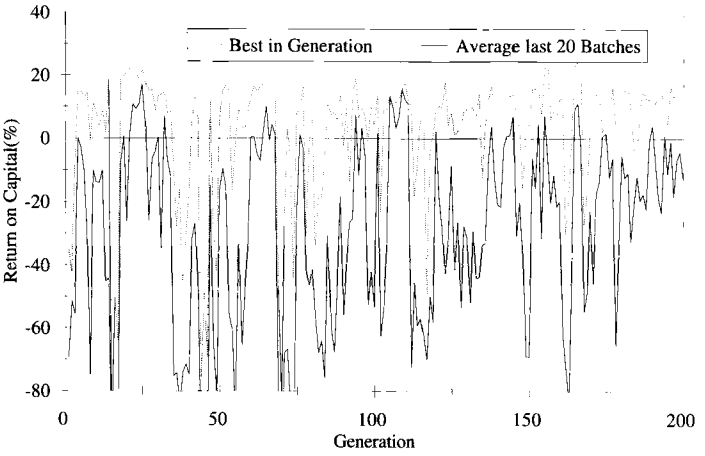


Figure 10. No GA, 20 new random rules each generation.

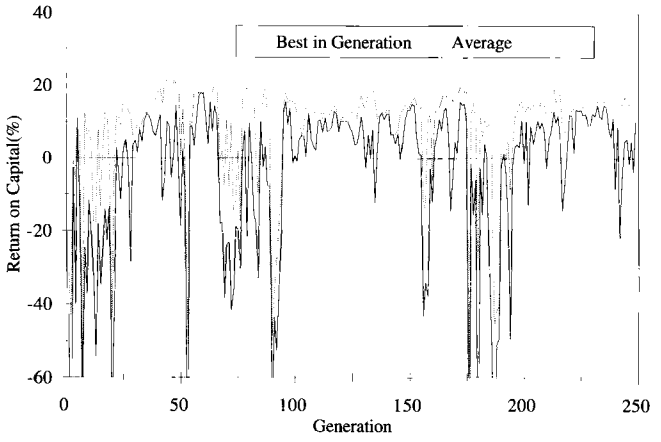


Figure 11. Short AOC learning, 20 batches per generation.

4.3 Performance, Problems, and Potential Improvements

That learning takes place is evident. As a comparative baseline, the results of controlling the simulation by generating random numbers gives average return of approximately −24% (Figure 12). Two possible weaknesses can be seen in the behavior of the CS.

4.3.1 Mistakes by Generalist Rules In the AOC phase very often the rule set obtaining the best result does not become the dominant rule set of the generation. (The probabilistic competition for rule selection will prevent the best rule set always becoming dominant but the best set dominates less often than might be expected.) This effect was noted in the case of the very long AOC (see Section 4.2). The bonus (for beating the best rule set so far) part of the reward scheme was introduced in an attempt to combat this.

The cause is almost certainly that generalization allows a rule that has built strength in appropriate situations, to win the bidding in a different situation where (because of its low specificity) its conditions match but its actions are inappropriate. Potential for this has

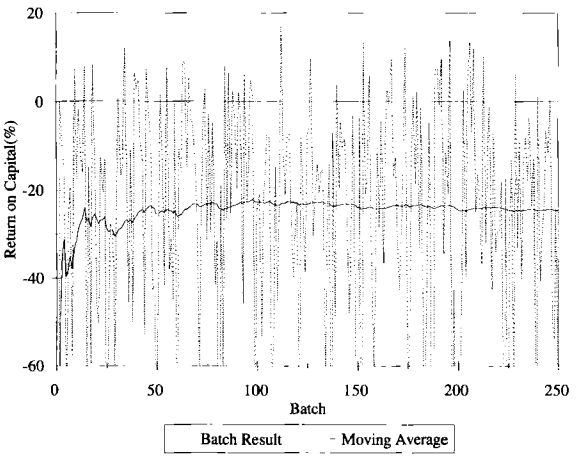


Figure 12. Random control.

been noted in the rules of Table 5 (see Section 4.1). The usual method for combating the mistakes of generalists is to scale the bidding to favor the specialist (Riollo, 1987) and this has been included from the first. In an attempt to further help the specialist, it was decided to distribute more reward to specialists but the benefit (if any) from this is not evident.

Possible strategies for improvement might be to (a) take advantage of and accentuate this generalization by grading the probability of a wild card in the rules such that the high-order bits are more likely to be specific. (This could be a useful practical technique but from the purist viewpoint it would mean that the CS has some knowledge of the significance of some of the genes!) or (b) institute a directed mutation in rules of high strength by which the most significant wild card may change to a specific bit that matches the most frequent use of that rule. (This might involve excessive record keeping.)

4.3.2 Inappropriate Crossover It is apparent that good results achieved in one generation can easily be lost in the GA when forming the next generation. Clearly this will occur when an inappropriate new chromosome of high strength is formed, for example from a good start-of-reaction parent and a good end-of-reaction parent by a crossover close to the condition-action boundary. The most promising, biologically analogous solution would appear to be to encourage niche (Holland, 1975; Cavicchio, 1970; Booker, 1985) populations for different periods of the reaction and to reduce the probability of interniche breeding. The "B added so far condition" could provide a tag for niche identification because this condition must increase through the reaction. The biological analogy would be organisms that exploit different concentrations of the products C and D and are less likely to breed with organisms that exist in other product concentrations.

4.3.3 Reaction Termination Control If niche populations can be established the disruptive crossover problems of a reaction termination action (see Section 2.3) would be simplified. This could be implemented as an action bit, always set to zero in the initial population, but with a chance of a special mutation when in presence of high product concentrations (high B added condition).

5. Conclusions

A CS has shown ability to learn the control of a production process and to optimize profit from that process. In particular it has been shown that the CS far outperforms control by randomly generated rules. That a GA is a valuable adjunct to AOC learning is also demonstrated by better profit than with AOC alone or AOC with replacement of weak rules by randomly generated rules. A degree of robustness has been demonstrated by its ability to adapt to changes in the market environment and in the process. In the instance of coping with high heats of reaction it learned to anticipate the heat rise by, when adding reactant (cause of the heat rise), simultaneously increasing reactor cooling; it also learned to keep within an upper temperature-limit restriction.

In its present state, direct use of this CS to control a *real* chemical plant is problematic. During learning the CS produces many batches that would be disastrous financially and potentially dangerous (depending on the chemicals and process involved). Improvements, via the niche mechanisms discussed above, will mitigate some of these problems. If a simulation exists for a real plant then clearly off-line training is possible; it is unlikely, however, that a simulation will reflect the real plant exactly, and there is the problem of what sort of genetic pool could be allowed, when the CS goes on-line, such that the CS could explore new regions it was led into by real-world fluctuations without going into disastrous areas. The development of a system where a model world is used alongside the real world to evaluate the CS suggestions and reject untenable ones may be worth exploration. No matter how well the CS may ultimately perform, its use would require a supervisory safety control system to exclude dangerous actions.

Future work will explore improvements via the niche mechanism and inclusion of a stop action, a comparative study of BB in place of PSP, and consideration of plant noise (e.g., sticking valves, fouled heat exchange surfaces) and market noise (e.g., market size, price fluctuation).

Acknowledgment

Thanks go to the referees for their constructive criticism and helpful suggestions.

References

Booker, L. B. (1982). *Intelligent behavior as an adaptation to the task environment.* (Doctoral dissertation, University of Michigan, Ann Arbor). *Dissertation Abstracts International, 43*(2), 496B.

Booker, L. B. (1985). Improving the performance of genetic algorithms in classifier systems. In J. J. Grefenstette (Ed.), *Proceedings of the First International Conference on Genetic Algorithms and Their Applications* (pp. 80–92). Pittsburgh, PA: Lawrence Erlbaum.

Cavicchio, D. J. (1970). *Adaptive search using simulated evolution.* Unpublished doctoral dissertation, University of Michigan, Ann Arbor.

Chen, Q., & Weigand, W. A. (1992). Neural net model of batch process optimization based on an extended genetic algorithm. *International Joint Conference on Neural Networks 1992, 4,* 519–524.

De Jong, K. A. (1975). An analysis of the behavior of a class of genetic adaptive systems (Doctoral dissertation, University of Michigan, 1975). *Dissertation Abstracts International, 36*(10), 5140B.

Frey, P. W., & Slate, D. J. (1991). Letter recognition using Holland-style adaptive classifiers. *Machine Learning, 6*(2), 161–182.

Goldberg, D. E. (1983). *Computer-aided gas pipeline design using genetic algorithms and rule learning.*

(Doctoral dissertation, University of Michigan, Ann Arbor). *Dissertation Abstracts International, 44*(10), 3174B.

Goldberg, D. E. (1989). *Genetic algorithms in search, optimization & machine learning.* Reading, MA: Addison-Wesley.

Goodloe, M., & Graves, S. J. (1988). Improving performance of an electric power expert system with genetic algorithms. *Proceedings of the First International Conference on the Applications of Artificial Intelligence and Expert Systems, IEA/AIE-88, 1,* 298–305.

Grefenstette, J. J. (1988). Credit assignment in rule discovery based on genetic algorithms, *Machine Learning, 3,* 225–245.

Holland, J. H. (1962). Outline for a logical theory of adaptive systems. *Journal of the Association for Computing Machinery, 3,* 297–314.

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

Holland, J. H. (1985). Properties of the bucket brigade. In J. J. Grefenstette (Ed.), *Proceedings of the First International Conference on Genetic Algorithms and Their Applications* (pp. 1–7). Pittsburgh, PA: Lawrence Erlbaum.

Holland, J. H., & Reitman, J. S. (1978). Cognitive systems based on adaptive algorithms. In D. A. Waterman & F. Hayes-Roth (Eds.), *Pattern directed inference systems* (pp. 313–329). New York: Academic Press.

Hollstien, R. B. (1971). *Artificial genetic adaptation in computer control systems.* (Doctoral dissertation, University of Michigan, 1971). *Dissertation Abstracts International, 23*(3), 1510B.

Katoh, N., Nakao, K., & Hanawa, M. (1989). Learning control of a batch reactor. *Computers in Chemical Engineering, 13,* 1273–1276.

Nisenfield, A. E., & Turk, M. A. (1986). Batch reactor control: Could an expert advisor help? *InTech,* (April), 57–64.

Rao, V. R., & Lee, W.-K. (1991). Neural networks approach to automatic startup and control of an exothermic batch reactor. *Proceedings of 1991 American Control Conference, 3,* 2854–2857.

Raed, A. A. Z., & Hassoun, M. H. (1993). Regulator control via genetic search assisted reinforcement. In S. Forrest (Ed.), *Proceedings of the Fifth International Conference on Genetic Algorithms* (pp. 254–260). San Mateo, CA: Morgan Kaufmann.

Regev, O., Lewin, D. R., & Lavie, R. (1989). Exothermic batch chemical reactor automation via expert system. *Dechema-Monographs, 116,* 71–80.

Riollo, R. L. (1987). Bucket brigade performance: Default hierarchies. In J. J. Grefenstette (Ed.), *Proceedings of the Second International Conference on Genetic Algorithms* (pp. 190–201). San Mateo, CA: Morgan Kaufmann.

Ryhiner, G., Dunn, I. J., Heinzle, E., & Rohani, S. (1992). Adaptive on-line optimal control of bioreactors: Application to anaerobic degradation. *Journal of Biotechnology, 22,* 89–106.