

Economic Modeling Using Evolutionary Algorithms: The Effect of a Binary Encoding of Strategies

Ludo Waltman, Nees Jan van Eck, Rommert Dekker,
and Uzay Kaymak

ERIM REPORT SERIES <i>RESEARCH IN MANAGEMENT</i>	
ERIM Report Series reference number	ERS-2009-028-LIS
Publication	May 2009
Number of pages	21
Persistent paper URL	http://hdl.handle.net/1765/16014
Email address corresponding author	lwaltman@ese.eur.nl
Address	Erasmus Research Institute of Management (ERIM) RSM Erasmus University / Erasmus School of Economics Erasmus Universiteit Rotterdam P.O.Box 1738 3000 DR Rotterdam, The Netherlands Phone: + 31 10 408 1182 Fax: + 31 10 408 9640 Email: info@erim.eur.nl Internet: www.erim.eur.nl

Bibliographic data and classifications of all the ERIM reports are also available on the ERIM website:
www.erim.eur.nl

REPORT SERIES
RESEARCH IN MANAGEMENT

ABSTRACT AND KEYWORDS	
Abstract	<p>We are concerned with evolutionary algorithms that are employed for economic modeling purposes. We focus in particular on evolutionary algorithms that use a binary encoding of strategies. These algorithms, commonly referred to as genetic algorithms, are popular in agent-based computational economics research. In many studies, however, there is no clear reason for the use of a binary encoding of strategies. We therefore examine to what extent the use of such an encoding may influence the results produced by an evolutionary algorithm. It turns out that the use of a binary encoding can have quite significant effects. Since these effects do not have a meaningful economic interpretation, they should be regarded as artifacts. Our findings indicate that in general the use of a binary encoding is undesirable. They also highlight the importance of employing evolutionary algorithms with a sensible economic interpretation.</p>
Free Keywords	<p>agent-based computational economics, evolutionary algorithm, genetic algorithm, binary encoding, premature convergence</p>
Availability	<p>The ERIM Report Series is distributed through the following platforms:</p> <p>Academic Repository at Erasmus University (DEAR), DEAR ERIM Series Portal</p> <p>Social Science Research Network (SSRN), SSRN ERIM Series Webpage</p> <p>Research Papers in Economics (REPEC), REPEC ERIM Series Webpage</p>
Classifications	<p>The electronic versions of the papers in the ERIM report Series contain bibliographic metadata by the following classification systems:</p> <p>Library of Congress Classification, (LCC) LCC Webpage</p> <p>Journal of Economic Literature, (JEL), JEL Webpage</p> <p>ACM Computing Classification System CCS Webpage</p> <p>Inspec Classification scheme (ICS), ICS Webpage</p>

Economic modeling using evolutionary algorithms: The effect of a binary encoding of strategies

Ludo Waltman Nees Jan van Eck Rommert Dekker
Uzay Kaymak

Econometric Institute, Erasmus School of Economics
Erasmus University Rotterdam

P.O. Box 1738, 3000 DR Rotterdam, The Netherlands

E-mail: {lwaltman,nvaneck,rdekker,kaymak}@ese.eur.nl

Abstract

We are concerned with evolutionary algorithms that are employed for economic modeling purposes. We focus in particular on evolutionary algorithms that use a binary encoding of strategies. These algorithms, commonly referred to as genetic algorithms, are popular in agent-based computational economics research. In many studies, however, there is no clear reason for the use of a binary encoding of strategies. We therefore examine to what extent the use of such an encoding may influence the results produced by an evolutionary algorithm. It turns out that the use of a binary encoding can have quite significant effects. Since these effects do not have a meaningful economic interpretation, they should be regarded as artifacts. Our findings indicate that in general the use of a binary encoding is undesirable. They also highlight the importance of employing evolutionary algorithms with a sensible economic interpretation.

Keywords

Agent-based computational economics, evolutionary algorithm, genetic algorithm, binary encoding, premature convergence.

1 Introduction

Evolutionary algorithms (EAs) are algorithms that are inspired by the process of natural evolution. EAs have their origins in the field of computer science, where they are mainly applied for optimization purposes. Nowadays, EAs are also regularly employed in the field of economics. In economic research, EAs frequently serve as a tool for modeling boundedly rational behavior. When EAs are applied as a modeling tool in economic research, a binary encoding of strategies is typically used. This means that strategies are represented by bit strings (i.e., strings of zeros and ones) and that evolutionarily inspired operations such as crossover and mutation take place at the level of individual bits. EAs that use a binary encoding of strategies are commonly referred to as genetic algorithms. For early research in which genetic algorithms are employed, we refer to Miller (1986, 1996), Axelrod (1987), Marks (1992), Arifovic (1994, 1996), Andreoni and Miller (1995), and Dawid (1996). Examples of more recent research can be found in the work of, among others, Lux and Schornstein (2005), Alkemade, La Poutré, and Amman (2006, 2007, 2009), Arifovic and Maschek (2006), Wheeler, Bean, Gaffney, and Taylor (2006), Xu (2006), and Casari (2008).

Researchers who apply genetic algorithms as a tool for modeling boundedly rational behavior typically do not justify why they use a binary encoding of strategies. If the agents whose behavior is being modeled have to make decisions that are intrinsically binary, such as decisions between cooperation and defection in a prisoner's dilemma (e.g., Axelrod, 1987), the use of a binary encoding of strategies is a very natural choice. However, in the case of non-binary decisions, such as decisions by firms on their production level (e.g., Arifovic, 1994; Price, 1997; Dawid & Kopel, 1998; Franke, 1998; Vriend, 2000; Alkemade et al., 2006, 2007, 2009; Arifovic & Maschek, 2006; Wheeler et al., 2006; Casari, 2008), there is no clear reason for the use of a binary encoding of strategies.

In this paper, we examine to what extent the use of a binary encoding of strategies may influence the results of studies in which EAs are employed. It turns out that the use of a binary encoding can have quite significant effects. In general, these effects do not have a meaningful economic interpretation and should be regarded as artifacts. In order to avoid these artifacts, we argue that in most cases researchers should not use a binary encoding of strategies.

Our research is inspired by results reported by Alkemade et al. (2006, 2007, 2009) (see also Waltman & Van Eck, 2009b). Alkemade et al. show that under certain conditions an EA

that is employed for modeling purposes may exhibit premature convergence. By premature convergence Alkemade et al. mean that different runs of the EA can lead to very different results. Alkemade et al. argue that premature convergence is caused by a too small population size. In this paper, we report results that point in a different direction. We show that the observation of premature convergence by Alkemade et al. depends crucially on their use of a binary encoding of strategies. Using the same economic environment as Alkemade et al. (i.e., a Cournot oligopoly market), we demonstrate that premature convergence does occur in the case of EAs with a binary encoding while it does not occur in the case of EAs without a binary encoding. These results indicate that the premature convergence observed by Alkemade et al. is an artifact caused by the use of a binary encoding of strategies.

We note that the consequences of the use of a binary encoding of strategies are also studied extensively by Dawid (1996) (see also Dawid & Kopel, 1998). However, the approach taken by Dawid is quite different from the approach that we take in the present paper. Dawid focuses on EAs with a large population size, and he is concerned with aggregate results, that is, results averaged over many EA runs. We do not assume the population size to be large, and we are specifically interested in comparing results of individual EA runs. Another difference is that the crossover operator plays a crucial role in Dawid's approach while in our approach the crossover operator is not important at all.

The remainder of this paper is organized as follows. In Sections 2 and 3, we discuss the Cournot market that we consider in this paper and we present the various EAs that we study. In Section 4, we report the results of the computer simulations that we have performed. Based on the results, we provide an elaborate analysis of the effect of a binary encoding of strategies in Section 5. Finally, in Section 6, we discuss the conclusions of our research.

2 Cournot oligopoly market

To analyze the effect of a binary encoding of strategies, we study the behavior of firms in a Cournot oligopoly market. We consider exactly the same Cournot market as Alkemade et al. (2006, 2007, 2009). For other studies in which quantity competition among firms is modeled using EAs, we refer to Arifovic (1994), Price (1997), Dawid and Kopel (1998), Franke (1998), Vriend (2000), Arifovic and Maschek (2006), Riechmann (2006), Wheeler et al. (2006), and

Casari (2008).

The Cournot market that we consider has the following characteristics: The number of firms equals four, firms produce perfect substitutes, the demand function is linear, firms have identical cost functions, and marginal cost is constant. The inverse demand function is given by

$$p = \max \left(256 - \sum_{i=1}^4 q_i, 0 \right), \quad (1)$$

where p denotes the market price and q_i denotes firm i 's production level. Firm i 's total cost equals $c_i = 56q_i$. Hence, it follows that firm i 's profit is given by

$$\pi_i = pq_i - c_i = q_i \max \left(200 - \sum_{i'=1}^4 q_{i'}, -56 \right). \quad (2)$$

A Nash (or Cournot) equilibrium is obtained if each firm chooses a production level that maximizes its profit given the production levels of its competitors. This means that in a Nash equilibrium $\partial\pi_i/\partial q_i = 0$ for $i = 1, \dots, 4$. It is easy to see that the Cournot market that we consider has a Nash equilibrium in which each firm produces a quantity of 40. Each firm makes a profit of 1600 in the Nash equilibrium. In addition to a Nash equilibrium, the Cournot market that we consider also has a competitive (or Walrasian) equilibrium. This equilibrium is obtained if firms are not aware of their influence on the market price and therefore behave as price takers. In the competitive equilibrium, the four firms jointly produce a quantity of 200 and each firm makes a profit of 0.

3 Evolutionary algorithms

As pointed out by Vriend (2000), there are two quite different ways in which EAs can be employed to model the behavior of economic agents. In the individual learning approach, each agent learns exclusively from its own experience (e.g., Arifovic, 1994; Price, 1997; Arifovic & Maschek, 2006; Casari, 2008). This is modeled through the use of a separate EA for each agent. In the social learning approach, each agent learns not only from its own experience but also from the experience of other agents (e.g., Arifovic, 1994; Dawid & Kopel, 1998; Franke, 1998; Alkemade et al., 2006, 2007, 2009; Riechmann, 2006). This is modeled through the use of a single EA for all agents together. The social learning approach seems to be more popular than the individual learning approach (Arifovic & Maschek, 2006). In this paper, we focus our attention on the social learning approach.

An important observation about the social learning approach is made by Alkemade et al. (2006, 2007, 2009). They note that the social learning approach can be implemented in two quite different ways. On the one hand, one can employ an EA with a population size that equals the number of interacting agents (e.g., Arifovic, 1994; Dawid & Kopel, 1998; Franke, 1998; Vriend, 2000; Riechmann, 2006). This results in a one-to-one relationship between strategies and agents. In the case of the Cournot oligopoly market discussed in the previous section, the EA would have a population size of four (since there are four firms in the market). On the other hand, one can employ an EA with a population size that exceeds the number of interacting agents (e.g., Axelrod, 1987; Andreoni & Miller, 1995; Miller, 1996; Dawid, 1996, Sections 4.5 and 5.3). Strategies are then evaluated using some matching mechanism. In the case of the Cournot oligopoly market discussed in the previous section, the EA would have a population size greater than four. Alkemade et al. show that the two ways in which the social learning approach can be implemented can lead to very different results.¹ In this paper, we only employ EAs with a population size that equals the number of interacting agents. We take this approach because it turns out to be very suitable for demonstrating how the use of a binary encoding of strategies can lead to artifacts.

We consider five different EAs in this paper. We refer to these EAs as EA1 to EA5. Each of the EAs provides a slightly different model of the behavior of firms in the Cournot market discussed in the previous section. The five EAs all have the same general form. This general form is shown in Figure 1. Each EA makes use of a population of four strategies. A strategy corresponds with the production level of one of the four firms in the Cournot market. The EAs all impose the constraint that the production level of a firm must lie between 0 and 127, and they all randomly generate an initial population by drawing four strategies from a uniform distribution over all possible strategies. In each iteration of an EA, the profit resulting from each of the four strategies in the current population is calculated using (2). Based on the profits of the four strategies, the fitness values of the strategies are calculated according to

$$f_i = \begin{cases} \max\left(\frac{\pi_i - \mu}{\sigma} + 2, 0\right) & \text{if } \sigma > 0, \\ 1 & \text{otherwise,} \end{cases} \quad (3)$$

¹In the evolutionary game theory literature, similar observations have been made by various researchers, for example by Hansen and Samuelson (1988) and Rhode and Stegeman (1996) in an economic context and by Schaffer (1988) in a biological context.

Randomly generate an initial population of strategies

repeat

 Calculate each strategy's profit

 Calculate each strategy's fitness

 Apply the selection operator

 Apply the crossover and mutation operators

until a specific number of iterations have been performed

Figure 1: General form of the five EAs considered in this paper.

where μ and σ denote, respectively, the mean and the standard deviation of the profits of the strategies, that is,

$$\mu = \frac{\sum_{i=1}^4 \pi_i}{4}, \quad (4)$$

$$\sigma = \sqrt{\frac{\sum_{i=1}^4 (\pi_i - \mu)^2}{4}}. \quad (5)$$

The above transformation from profits to fitness values is sometimes referred to as sigma scaling (e.g., Mitchell, 1996) or sigma truncation (e.g., Goldberg, 1989). The transformation is used by, for example, Axelrod (1987), Andreoni and Miller (1995), Miller (1996), and Franke (1998). The five EAs that we study all use roulette wheel selection (e.g., Goldberg, 1989; Mitchell, 1996), also known as fitness-proportionate selection. This means that in each iteration of an EA the selection operator generates a new population of strategies by randomly drawing four strategies from the old population. Strategies are drawn independently and with replacement. The probability that a strategy is drawn is proportional to the fitness of the strategy given by (3).

The five EAs that we consider differ from each other on the following four dimensions:

1. The type of strategies that are used, that is, integer production levels or real-valued production levels.
2. Whether a binary encoding of strategies is used or not.
3. Whether a crossover operator is used or not.
4. The type of mutation operator that is used.

The differences between the EAs are summarized in Table 1. We now discuss the specific characteristics of each of the EAs.

Table 1: Overview of the differences between the five EAs considered in this paper.

	EA1	EA2	EA3	EA4	EA5
Strategy	integer	integer	integer	integer	real number
Binary encoding	yes	yes	no	no	no
Crossover	yes	no	no	no	no
Mutation	bit flip	bit flip	uniform dist.	plus/minus 1	normal dist.

3.1 EA1

EA1 is a standard genetic algorithm (e.g., Goldberg, 1989; Mitchell, 1996). It uses integer production levels and a binary encoding of strategies. A strategy is represented by a bit string of length seven. The production level corresponding to a bit string (b_1, \dots, b_7) , where $b_j \in \{0, 1\}$ denotes the value of the j th bit in the string, is given by

$$q = \sum_{j=1}^7 2^{7-j} b_j. \quad (6)$$

EA1 uses a single-point crossover operator (e.g., Mitchell, 1996) with a crossover rate of 1. The mutation operator used by EA1 randomly determines for each bit in a bit string whether the bit should be inverted or not. The mutation rate equals 0.001, which means that each bit has a probability of 0.001 of being inverted. We note that EA1 is very similar to the EA employed by Alkemade et al. (2006, 2007, 2009).

3.2 EA2

EA2 is identical to EA1 except that it does not use a crossover operator.

3.3 EA3

Like EA2, EA3 uses integer production levels and does not use a crossover operator. Unlike EA2, EA3 does not use a binary encoding of strategies. EA3 also uses a different mutation operator than EA2. The probability that a strategy is being mutated equals 0.01. If a strategy is being mutated, it is replaced by a random new strategy that is drawn from a uniform distribution over all possible strategies. In the economic literature, EAs similar to EA3 are employed by Ünver (2001), Dawid and Dermietzel (2006), and Haruvy, Roth, and Ünver (2006).

3.4 EA4

EA4 is identical to EA3 except that it uses a different mutation operator. The probability that a strategy is being mutated equals 0.01. If a strategy is being mutated, the corresponding production level is either increased by one or decreased by one (both with a probability of 0.5). The increase or decrease does not take place if the resulting new production level would be below 0 or above 127.

3.5 EA5

Like EA3 and EA4, EA5 does not use a binary encoding of strategies and also does not use a crossover operator. Unlike EA3 and EA4, EA5 uses real-valued production levels. EA5 also uses a different mutation operator than EA3 and EA4. The probability that a strategy is being mutated equals 0.01. If a strategy is being mutated, the corresponding production level is updated according to

$$q_{\text{new}} = \min \left(\max \left(q_{\text{old}} + N(0, s^2), 0 \right), 127 \right), \quad (7)$$

where q_{old} denotes the production level before mutation, q_{new} denotes the production level after mutation, and $N(0, s^2)$ denotes a normally distributed random variable with mean 0 and standard deviation s . EA5 uses a value of 1 for the parameter s . We note that EA5 is somewhat similar to what is referred to as an evolution strategy in the computer science literature (e.g., Beyer, 2001; Beyer & Schwefel, 2002). In the economic literature, EAs similar to EA5 are employed by Sellgren (2001), Gerding, Van Bragt, and La Poutre (2003), Lux and Schornstein (2005), Clemens and Riechmann (2006), and Riechmann (2006).

4 Simulation results

In this section, we report the results of the computer simulations that we have performed.² Each of the five EAs discussed in the previous section was run 100 times, each time using different random numbers. Each run lasted 10 000 iterations. We note that the results reported below are fairly robust to changes in the values of the various EA parameters. By changing parameter

²The software used to obtain the results is available online at <http://www.ludowaltman.nl/binaryencoding/>. The software runs in MATLAB.

values, somewhat different results may be obtained, but the analysis will remain essentially unchanged. The results reported below are also robust to changes in the transformation from profits to fitness values. We further experimented with simulation runs that lasted one million instead of 10 000 iterations, but this also did not affect the analysis in any fundamental way.

The results of six selected runs of EA1 are shown in Figure 2. Each graph in the figure corresponds with one run of EA1. The graphs display how the average production level of the four firms in the market evolves over time. In the graphs in panels (a), (b), and (c), firms' average production level stabilizes fairly quickly, respectively at a quantity of 47, 50, and 64. These results are quite typical. In almost all 100 runs of EA1, we observe that firms' average production level stabilizes at one of these three quantities. However, firms' average production level does not always remain stabilized at the same quantity during an entire run. In a few runs of EA1, we find that firms' average production level switches from one stable quantity to another. Panels (d), (e), and (f) of Figure 2 provide examples of such runs.

In panel (a) of Figure 3, the results of all 100 runs of EA1 are averaged. As can be seen, on average firms' production level stabilizes at a quantity of about 52. Based on the 100 runs of EA1, the distribution of firms' average production level at the end of a run can be determined. This distribution is displayed in panel (b) of Figure 3. It turns out that in somewhat more than half of the runs firms' average production level after 10 000 iterations equals the stable quantity of 50. In most other runs, firms' average production level after 10 000 iterations equals either the stable quantity of 47 or the stable quantity of 64. There are a few runs in which firms' average production level after 10 000 iterations does not equal one of the three stable quantities. In most of these runs, this is probably due to small disturbances caused by the mutation operator.

The results that we have obtained using EA1 are very similar to the results reported by Alkemade et al. (2006, 2007, 2009).³ Like Alkemade et al., we find that in different EA runs firms' average production level stabilizes at different quantities. This phenomenon is referred to as premature convergence by Alkemade et al.. We further find that on average firms' production level stabilizes at a quantity above 50. This means that on average firms produce a larger quantity than in the competitive equilibrium of the Cournot oligopoly market (see Section 2). This finding is also in agreement with the results reported by Alkemade et al.

³Due to an error in their computer simulations, the results reported by Alkemade et al. (2006, 2007) are not entirely correct. For a correction of the results, see Alkemade et al. (2009). For some additional comments on the results, see Waltman and Van Eck (2009b).

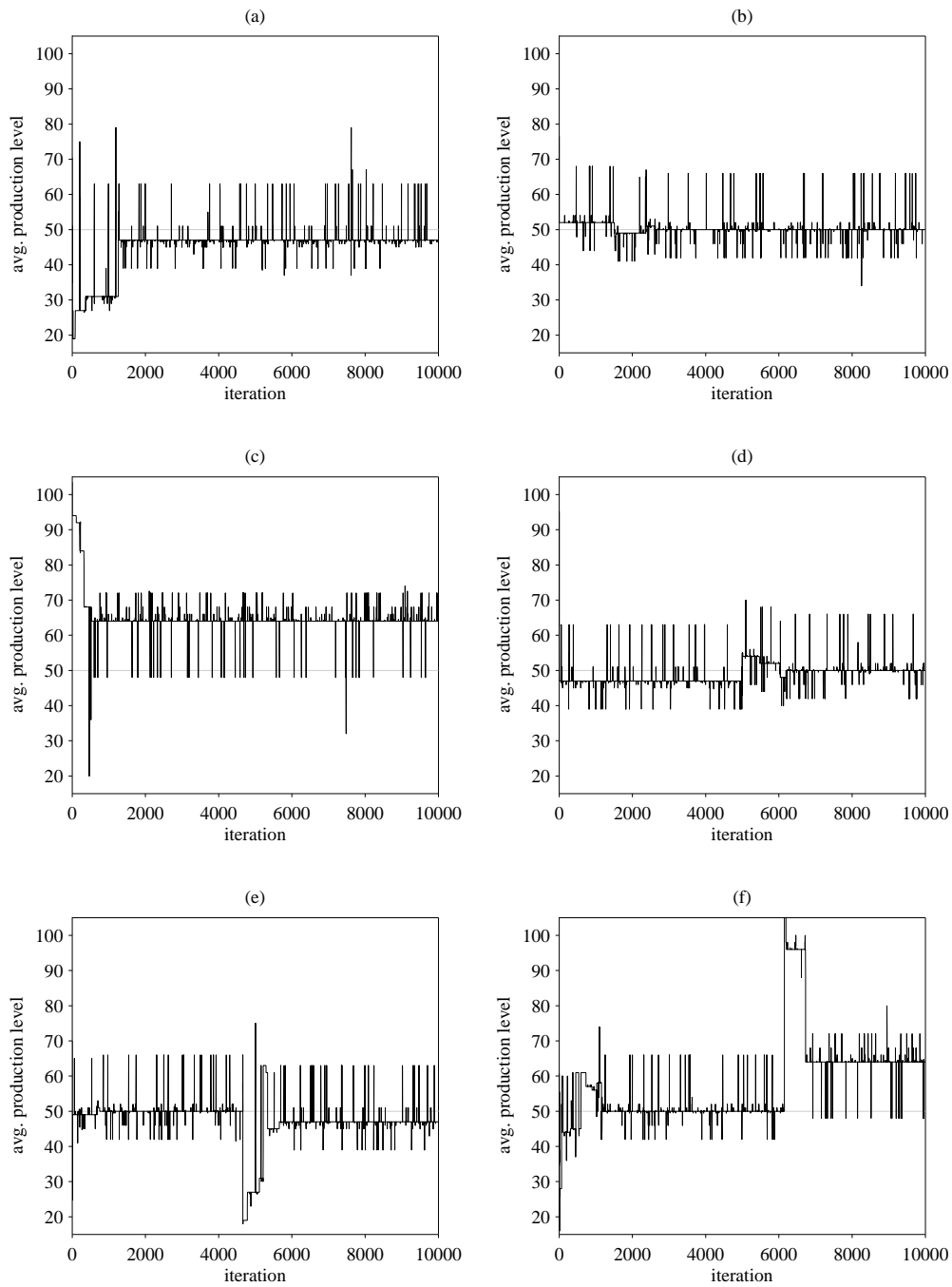


Figure 2: Results of six selected runs of EA1. The graphs display for each run how firms' average production level evolves over time. The horizontal line in each graph indicates the competitive equilibrium quantity of 50.

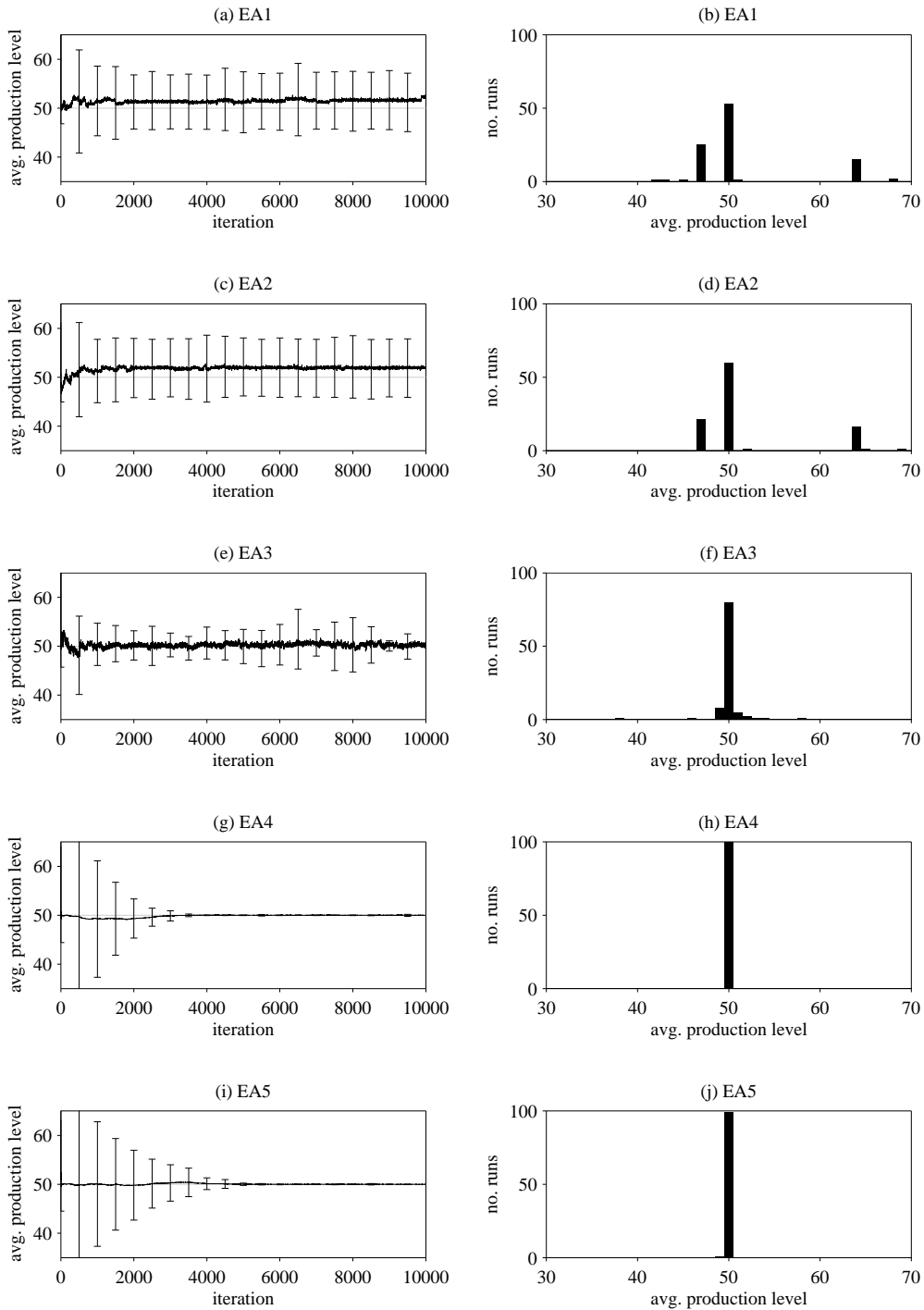


Figure 3: Results of EA1 to EA5. The graphs in the left panels display for each EA how firms' average production level evolves over time. The results shown in the graphs are averages over 100 EA runs. Error bars indicate standard deviations. The horizontal line in each graph indicates the competitive equilibrium quantity of 50. The histograms in the right panels display for each EA the distribution of firms' average production level at the end of a run. The distributions are based on 100 EA runs.

We now turn to EA2. The results obtained using EA2 are shown in panels (c) and (d) of Figure 3. It is clear that the results of EA2 are quite similar to the results of EA1. Like EA1, EA2 leads to premature convergence. Firms' average production level again stabilizes at a quantity of 47, 50, or 64. The similarity between the results of EA1 and EA2 is not surprising. The only difference between the two EAs is that EA1 uses a crossover operator while EA2 does not use such an operator. In an earlier paper (Waltman & Van Eck, 2009a), we have shown mathematically that, if the mutation rate is small and some technical assumptions are satisfied, the effect of the use of a crossover operator on the results produced by an EA tends to be negligible in the long run. The results shown in Figure 3 are in line with this theoretical finding.

Finally, we consider EA3, EA4, and EA5. The results obtained using these EAs are shown in panels (e) to (j) of Figure 3. As can be seen, in the long run the three EAs produce quite similar results. The only noteworthy difference is that the results of EA3 are more volatile than the results of EA4 and EA5. However, this is to be expected, since EA3 uses a more disruptive mutation operator than EA4 and EA5 (see Section 3). What is much more interesting to look at is the difference between the results of EA3, EA4, and EA5 on the one hand and the results of EA1 and EA2 on the other hand. When looking at the aggregate results of 100 EA runs (see the left panels of Figure 3), it can be seen that firms' average production level stabilizes at a quantity of 50 in the case of EA3, EA4, and EA5 while it stabilizes at a quantity of about 52 in the case of EA1 and EA2. Hence, in the case of EA3, EA4, and EA5, firms on average produce the quantity associated with the competitive equilibrium of the Cournot market (see Section 2). In the case of EA1 and EA2, on the other hand, firms on average produce a quantity that is larger than the competitive equilibrium quantity. When looking at the results of individual EA runs (see the right panels of Figure 3), it turns out that firms' average production level stabilizes around a quantity of 50 in the case of EA3, EA4, and EA5 while it stabilizes at a quantity of 47, 50, or 64 in the case of EA1 and EA2. Hence, premature convergence only takes place in the case of EA1 and EA2. It does not take place in the case of EA3, EA4, and EA5. Based on the above observations, it is clear that, both at the aggregate level and at the level of individual runs, EA3, EA4, and EA5 produce fundamentally different results than EA1 and EA2.

The principal difference between EA1 and EA2 on the one hand and EA3, EA4, and EA5 on the other hand is that EA1 and EA2 use a binary encoding of strategies while EA3, EA4, and EA5 do not use such an encoding (see Table 1). Based on the results reported in this section,

it seems that the use of a binary encoding of strategies can have quite significant effects. In the next section, we provide an analysis of these effects, in particular of the phenomenon of premature convergence.

5 Analysis of the effect of a binary encoding of strategies

For each of the five EAs considered in the previous section, it turned out that in at least a substantial number of runs firms' average production level stabilized at a quantity of 50, that is, at the quantity associated with the competitive equilibrium of the Cournot oligopoly market. There is a straightforward explanation for this finding. Suppose that in some iteration of an EA the population consists of four identical strategies, each corresponding with a production level of 50. The market price then equals firms' constant marginal cost, and each strategy therefore results in a profit of 0. Suppose now that the mutation operator changes one of the strategies in the population. We refer to this strategy as strategy A. Strategy A's production level may either increase or decrease. If strategy A's production level increases, the total quantity produced in the market will increase and, as a consequence, the market price will fall below firms' constant marginal cost. All four strategies in the population will then result in a loss, but strategy A will result in a larger loss than the other strategies. If on the other hand strategy A's production level decreases, the total quantity produced in the market will decrease and, as a consequence, the market price will rise above firms' constant marginal cost. All four strategies in the population will then result in a profit, but strategy A will result in a smaller profit than the other strategies. Hence, regardless of whether strategy A's production level increases or decreases, the fitness of strategy A will always be lower than the fitness of the other strategies in the population. As a consequence, the probability that in the next iterations of the EA strategy A remains in the population is quite low. Most likely, within one or a few iterations, the selection operator will remove strategy A from the population. The population will then return to its original state, that is, it will again consist of four identical strategies, each corresponding with a production level of 50.

The above mechanism explains why in the EAs considered in the previous section firms' average production level tends to stabilize at the competitive equilibrium quantity of 50. It should be noted that the above mechanism has been discussed quite extensively in the evolutionary

game theory literature. The mechanism was first discussed by Hansen and Samuelson (1988) and Schaffer (1989), and a comprehensive mathematical treatment of the mechanism was provided by Vega-Redondo (1997). A discussion of the mechanism can also be found in a study by Vriend (2000) on EA modeling in a Cournot oligopoly environment.

As discussed in the previous section, the fundamental difference between the results obtained using EA1 and EA2 on the one hand and the results obtained using EA3, EA4, and EA5 on the other hand is that in the case of EA1 and EA2 firms' average production level can stabilize not only at a quantity of 50 but also at a quantity of 47 or 64. This phenomenon of multiple stable quantities is referred to as premature convergence by Alkemade et al. (2006, 2007, 2009). We now show that the premature convergence phenomenon is caused by the use of a binary encoding of strategies.

Suppose that in some iteration of EA1 or EA2 the population consists of four identical strategies, each corresponding with a production level of 47. Each strategy is then represented by the bit string 0101111. Suppose now that out of the 28 bits used to represent the four strategies in the population exactly one bit is inverted by the mutation operator. This means that after applying the mutation operator one of the four strategies in the population has changed while the other three strategies have not changed. We refer to the strategy that has changed as strategy A. The first column of Table 2 lists seven bit strings. It is clear that one of these bit strings must represent strategy A (which one depends on which bit has been inverted). For each bit string, the corresponding production level is listed in the second column of the table. The last two columns of the table list for each production level the resulting profit of strategy A as well as the resulting profit of the other three strategies in the population, that is, the strategies corresponding with a production level of 47. Profits were calculated using (1) and (2). As can be seen in the table, the profit of strategy A will always be smaller than the profit of the other three strategies, regardless of which bit has been inverted. This means that within one or a few iterations of the EA the selection operator will most likely remove strategy A from the population. The population will then return to its original state, that is, it will again consist of four identical strategies, each corresponding with a production level of 47.

In the case of EA1 and EA2, the above mechanism shows that, if the population is in a state in which each strategy corresponds with a production level of 47, the inversion of a single bit is unlikely to upset this state for more than a few iterations. Of course, things may be different

Table 2: Effect of the inversion of a single bit given a population in which each strategy corresponds with a production level of 47 (represented by the bit string 0101111).

Bit string strategy A	Production level strategy A	Profit strategy A	Profit other strategies
0101110	46	598	611
0101101	45	630	658
0101011	43	688	752
0100111	39	780	940
0111111	63	-252	-188
0001111	15	660	2068
1101111	111	-5772	-2444

when the mutation operator inverts two or more bits at the same time. However, this happens only very rarely. (Given a mutation rate of 0.001, this happens on average once in every 2692 iterations of an EA.) The above mechanism therefore explains why in the case of EA1 and EA2 firms' average production level can stabilize at a quantity of 47.

The same explanation also holds for a quantity of 64. This quantity corresponds with the bit string 1000000. The bit strings that can be obtained by inverting a single bit are listed in the first column of Table 3. This table has a similar structure as Table 2. Like in Table 2, the profits listed in the third column of Table 3 are always smaller than those listed in the fourth column. This indicates that, if the population is in a state in which each strategy corresponds with a production level of 64, the inversion of a single bit is unlikely to upset this state for more than a few iterations. Taking into account that the simultaneous inversion of two or more bits happens only very rarely, this explains why 64 is a stable quantity in the case of EA1 and EA2.

A question that remains is whether in the case of EA1 and EA2 there are other stable quantities in addition to 47, 50, and 64. To answer this question, we calculated tables similar to Tables 2 and 3 for all integer quantities between 0 and 127. It turned out that 47, 50, and 64 are the only quantities for which the inversion of a single bit always results in a smaller profit for the mutated strategy than for the three non-mutated strategies. 47, 50, and 64 are therefore the only stable quantities. All other quantities are unstable. Consider for example Table 4. This table was calculated for a quantity of 48. As can be seen in the table, the inversion of one of

Table 3: Effect of the inversion of a single bit given a population in which each strategy corresponds with a production level of 64 (represented by the bit string 1000000).

Bit string strategy A	Production level strategy A	Profit strategy A	Profit other strategies
1000001	65	-3640	-3584
1000010	66	-3696	-3584
1000100	68	-3808	-3584
1001000	72	-4032	-3584
1010000	80	-4480	-3584
1100000	96	-5376	-3584
0000000	0	0	512

the three rightmost bits results in a larger profit for the mutated strategy (referred to as strategy A in the table) than for the three non-mutated strategies. This indicates that, given a population in which each strategy corresponds with a production level of 48, the inversion of a single bit can relatively easily trigger a transition to a completely different population. This makes 48 an unstable quantity.

We have now shown how the use of a binary encoding of strategies causes the premature convergence observed in the case of EA1 and EA2. Based on our analysis, it is clear that the phenomenon of premature convergence depends crucially on the use of a binary encoding of strategies. We note that our analysis is perfectly in line with the simulation results reported in the previous section. The analysis not only explains why premature convergence occurs specifically in the case of EA1 and EA2, but it also explains why the quantities of 47, 50, and 64 are stable and all other quantities are unstable. In this way, the analysis provides a good understanding of the most important results shown in Figures 2 and 3.

6 Conclusions

In a paper on EA modeling, Dawid and Kopel (1998) warn that “we have to be aware of the fact that simulation results may crucially depend on implementation details which have hardly any economic meaning” (p. 311). The present paper can be seen as an illustration of this im-

Table 4: Effect of the inversion of a single bit given a population in which each strategy corresponds with a production level of 48 (represented by the bit string 0110000).

Bit string strategy A	Production level strategy A	Profit strategy A	Profit other strategies
0110001	49	343	336
0110010	50	300	288
0110100	52	208	192
0111000	56	0	0
0100000	32	768	1152
0010000	16	640	1920
1110000	112	-6272	-2688

portant but somewhat overlooked point. In the context of quantity competition among firms, it is difficult if not impossible to give a sensible economic interpretation to the use of a binary encoding of strategies. In fact, the use of a binary encoding seems merely a relic from the genetic algorithm literature in the field of computer science. Of course, nothing would be wrong with the use of a binary encoding if its effect on the results produced by an EA were insignificant. However, our computer simulations and the subsequent analysis make clear that this need not be the case. They show that the use of a binary encoding may lead to a phenomenon known as premature convergence. This phenomenon is an artifact that depends crucially on strategies being encoded in binary form.

Based on our findings, we conclude that in general the use of a binary encoding of strategies is undesirable. By not using a binary encoding, one avoids the risk of having to deal with all kinds of artifacts, such as the premature convergence observed by Alkemade et al. (2006, 2007, 2009). For various examples of studies in which EAs are employed without using a binary encoding, we refer to Sellgren (2001), Ünver (2001), Gerding et al. (2003), Lux and Schornstein (2005), Clemens and Riechmann (2006), Dawid and Dermietzel (2006), Haruvy et al. (2006), and Riechmann (2006). It should be noted, however, that there are special cases in which we consider the use of a binary encoding perfectly acceptable. In a prisoner's dilemma, for example, agents have to make decisions that are intrinsically binary, namely decisions between cooperation and defection. The use of a binary encoding of strategies (e.g., Axelrod, 1987) then

seems a very natural choice that is unlikely to cause any artifacts.

The more general point that we want to make is that, when one employs an EA for economic modeling, all elements of the EA should have a meaningful economic interpretation (see also Dawid & Dermietzel, 2006).⁴ Many EAs employed in economic research have been adopted from the computer science literature without any substantial modification. Such EAs are likely to contain elements of which the economic interpretation is unclear. The use of a binary encoding of strategies is an example of such an element. Other EA elements of which the economic interpretation requires special attention include the population size (Alkemade et al., 2006, 2007, 2009), the selection operator (Van Bragt, Van Kemenade, & La Poutré, 2001; Dawid & Dermietzel, 2006), and the strategy updating regime (Dawid & Dermietzel, 2006). As we have shown in this paper, EA elements without a sensible economic interpretation may lead to simulation results that lack a sound underlying economic rationale. To avoid such results, paying close attention to the economic interpretation of the various elements of an EA is absolutely essential.

References

- Alkemade, F., La Poutré, H., & Amman, H. M. (2006). Robust evolutionary algorithm design for socio-economic simulation. *Computational Economics*, 28(4), 355–370.
- Alkemade, F., La Poutré, H., & Amman, H. M. (2007). On social learning and robust evolutionary algorithm design in the Cournot oligopoly game. *Computational Intelligence*, 23(2), 162–175.
- Alkemade, F., La Poutré, H., & Amman, H. M. (2009). Robust evolutionary algorithm design for socio-economic simulation: A correction. *Computational Economics*, 33(1), 99–101.
- Andreoni, J., & Miller, J. H. (1995). Auctions with artificial adaptive agents. *Games and Economic Behavior*, 10(1), 39–64.
- Arifovic, J. (1994). Genetic algorithm learning and the cobweb model. *Journal of Economic Dynamics and Control*, 18(1), 3–28.

⁴This point stands in stark contrast with one of the conclusions reached by Alkemade et al. (2006, 2007, 2009). They state that “economic model parameters and evolutionary algorithm parameters should be treated separately” (Alkemade et al., 2006, p. 367).

- Arifovic, J. (1996). The behavior of the exchange rate in the genetic algorithm and experimental economies. *Journal of Political Economy*, 104(3), 510–541.
- Arifovic, J., & Maschek, M. K. (2006). Revisiting individual evolutionary learning in the cobweb model: An illustration of the virtual spite-effect. *Computational Economics*, 28(4), 333–354.
- Axelrod, R. (1987). The evolution of strategies in the iterated prisoner's dilemma. In L. Davis (Ed.), *Genetic algorithms and simulated annealing* (pp. 32–41). Morgan Kaufmann.
- Beyer, H.-G. (2001). *The theory of evolution strategies*. Springer.
- Beyer, H.-G., & Schwefel, H.-P. (2002). Evolution strategies: A comprehensive introduction. *Natural Computing*, 1(1), 3–52.
- Casari, M. (2008). Markets in equilibrium with firms out of equilibrium: A simulation study. *Journal of Economic Behavior and Organization*, 65(2), 261–276.
- Clemens, C., & Riechmann, T. (2006). Evolutionary dynamics in public good games. *Computational Economics*, 28(4), 399–420.
- Dawid, H. (1996). *Adaptive learning by genetic algorithms: Analytical results and applications to economic models* (No. 441). Springer.
- Dawid, H., & Dermietzel, J. (2006). How robust is the equal split norm? Responsive strategies, selection mechanisms and the need for economic interpretation of simulation parameters. *Computational Economics*, 28(4), 371–397.
- Dawid, H., & Kopel, M. (1998). On economic applications of the genetic algorithm: A model of the cobweb type. *Journal of Evolutionary Economics*, 8(3), 297–315.
- Franke, R. (1998). Coevolution and stable adjustments in the cobweb model. *Journal of Evolutionary Economics*, 8(4), 383–406.
- Gerding, E., Van Bragt, D., & La Poutré, H. (2003). Multi-issue negotiation processes by evolutionary simulation, validation and social extensions. *Computational Economics*, 22(1), 39–63.
- Goldberg, D. E. (1989). *Genetic algorithms in search, optimization, and machine learning*. Addison-Wesley.
- Hansen, R. G., & Samuelson, W. F. (1988). Evolution in economic games. *Journal of Economic Behavior and Organization*, 10(3), 315–338.
- Haruvy, E., Roth, A. E., & Ünver, M. U. (2006). The dynamics of law clerk matching: An ex-

- perimental and computational investigation of proposals for reform of the market. *Journal of Economic Dynamics and Control*, 30(3), 457–486.
- Lux, T., & Schornstein, S. (2005). Genetic learning as an explanation of stylized facts of foreign exchange markets. *Journal of Mathematical Economics*, 41(1–2), 169–196.
- Marks, R. E. (1992). Breeding hybrid strategies: Optimal behaviour for oligopolists. *Journal of Evolutionary Economics*, 2(1), 17–38.
- Miller, J. H. (1986). *A genetic model of adaptive economic behavior* (Working paper). University of Michigan.
- Miller, J. H. (1996). The coevolution of automata in the repeated prisoner’s dilemma. *Journal of Economic Behavior and Organization*, 29(1), 87–112.
- Mitchell, M. (1996). *An introduction to genetic algorithms*. MIT Press.
- Price, T. C. (1997). Using co-evolutionary programming to simulate strategic behaviour in markets. *Journal of Evolutionary Economics*, 7(3), 219–254.
- Rhode, P., & Stegeman, M. (1996). A comment on “learning, mutation, and long-run equilibria in games”. *Econometrica*, 64(2), 443–449.
- Riechmann, T. (2006). Cournot or Walras? Long-run results in oligopoly games. *Journal of Institutional and Theoretical Economics*, 162(4), 702–720.
- Schaffer, M. E. (1988). Evolutionarily stable strategies for a finite population and a variable contest size. *Journal of Theoretical Biology*, 132(4), 469–478.
- Schaffer, M. E. (1989). Are profit-maximisers the best survivors? A Darwinian model of economic natural selection. *Journal of Economic Behavior and Organization*, 12(1), 29–45.
- Sellgren, A. (2001). The evolution of insurance markets under adverse selection. *Journal of Evolutionary Economics*, 11(5), 501–526.
- Ünver, M. U. (2001). Backward unraveling over time: The evolution of strategic behavior in the entry level British medical labor markets. *Journal of Economic Dynamics and Control*, 25(6–7), 1039–1080.
- Van Bragt, D. D. B., Van Kemenade, C., & La Poutré, H. (2001). The influence of evolutionary selection schemes on the iterated prisoner’s dilemma. *Computational Economics*, 17(2–3), 253–263.
- Vega-Redondo, F. (1997). The evolution of Walrasian behavior. *Econometrica*, 65, 375–384.

- Vriend, N. J. (2000). An illustration of the essential difference between individual and social learning, and its consequences for computational analyses. *Journal of Economic Dynamics and Control*, 24(1), 1–19.
- Waltman, L., & Van Eck, N. J. (2009a). *A mathematical analysis of the long-run behavior of genetic algorithms for social modeling* (Working paper No. ERS-2009-011-LIS). Erasmus University Rotterdam, Erasmus Research Institute of Management.
- Waltman, L., & Van Eck, N. J. (2009b). Robust evolutionary algorithm design for socio-economic simulation: Some comments. *Computational Economics*, 33(1), 103–105.
- Wheeler, S., Bean, N., Gaffney, J., & Taylor, P. (2006). A Markov analysis of social learning and adaptation. *Journal of Evolutionary Economics*, 16(3), 299–319.
- Xu, Y. (2006). The behavior of the exchange rate in the genetic algorithm with agents having long memory. *Journal of Evolutionary Economics*, 16(3), 279–297.

Publications in the Report Series Research* in Management

ERIM Research Program: "Business Processes, Logistics and Information Systems"

2009

How to Normalize Co-Occurrence Data? An Analysis of Some Well-Known Similarity Measures

Nees Jan van Eck and Ludo Waltman

ERS-2009-001-LIS

<http://hdl.handle.net/1765/14528>

Spare Parts Logistics and Installed Base Information

Muhammad N. Jalil, Rob A. Zuidwijk, Moritz Fleischmann, and Jo A.E.E. van Nunen

ERS-2009-002-LIS

<http://hdl.handle.net/1765/14529>

Open Location Management in Automated Warehousing Systems

Yugang YU and René B.M. de Koster

ERS-2009-004-LIS

<http://hdl.handle.net/1765/14615>

VOSviewer: A Computer Program for Bibliometric Mapping

Nees Jan van Eck and Ludo Waltman

ERS-2009-005-LIS

<http://hdl.handle.net/1765/14841>

Nash Game Model for Optimizing Market Strategies, Configuration of Platform Products in a Vendor Managed Inventory (VMI) Supply Chain for a Product Family

Yugang Yu and George Q. Huang

ERS-2009-009-LIS

<http://hdl.handle.net/1765/15029>

A Mathematical Analysis of the Long-run Behavior of Genetic Algorithms for Social Modeling

Ludo Waltman and Nees Jan van Eck

ERS-2009-011-LIS

<http://hdl.handle.net/1765/15181>

A Taxonomy of Bibliometric Performance Indicators Based on the Property of Consistency

Ludo Waltman and Nees Jan van Eck

ERS-2009-014-LIS

<http://hdl.handle.net/1765/15182>

A Stochastic Dynamic Programming Approach to Revenue Management in a Make-to-Stock Production System

Rainer Quante, Moritz Fleischmann, and Herbert Meyr

ERS-2009-015-LIS

<http://hdl.handle.net/1765/15183>

Some Comments on Egghe's Derivation of the Impact Factor Distribution

Ludo Waltman and Nees Jan van Eck

ERS-2009-016-LIS

<http://hdl.handle.net/1765/15184>

The Value of RFID Technology Enabled Information to Manage Perishables

Michael Ketzenberg, and Jacqueline Bloemhof

ERS-2009-020-LIS

<http://hdl.handle.net/1765/15412>

The Environmental Gains of Remanufacturing: Evidence from the Computer and Mobile Industry
J. Quariguasi Frota Neto, and J.M. Bloemhof
ERS-2009-024-LIS
<http://hdl.handle.net/1765/15912>

Economic Modeling Using Evolutionary Algorithms: The Effect of a Binary Encoding of Strategies
Ludo Waltman, Nees Jan van Eck, Rommert Dekker, and Uzay Kaymak
ERS-2009-028-LIS
<http://hdl.handle.net/1765/16014>

* A complete overview of the ERIM Report Series Research in Management:
<https://ep.eur.nl/handle/1765/1>

ERIM Research Programs:
LIS Business Processes, Logistics and Information Systems
ORG Organizing for Performance
MKT Marketing
F&A Finance and Accounting
STR Strategy and Entrepreneurship