Journal of Mathematics and System Science 8 (2018) 25-43
doi: 10.17265/2159-5291/2018.02.001

Social Simulation for Analyzing Product Recall Systems Using Co-Evolution Model with Price Competition

TetsurohWatanabe[1], Taro Kanno[2], and Kazuo Furuta[3]

1. *Department of Technology Management for Innovation, School of Engineering, The University of Tokyo*
   7-3-1 Hongo, Bunkyo-ku, Tokyo, 113-8656, Japan
2. *Department of Systems Innovation, School of Engineering, The University of Tokyo*
   7-3-1 Hongo, Bunkyo-ku, Tokyo, 113-8656, Japan
3. *Department of Technology Management for Innovation, School of Engineering, The University of Tokyo*
   7-3-1 Hongo, Bunkyo-ku, Tokyo, 113-8656, Japan

Abstract: In recent years, accidents and product recalls caused by product defects have become important problems in numerous industries worldwide. Nevertheless, most existing studies have examined product recalls using empirical approaches. To improve product recall systems, we studied social simulation using a multi-agent system with a co-evolution model. This research is important because empirical approaches are no longer adequate for complex and diverse modern societies. Discussions using quantitative and predictive approaches, including agent-based simulation, are therefore expected. For this study, we used a Layered Co-evolution Model to reflect situations of the real society using producer agents and consumer agents. Additionally, we applied multi-objective optimization techniques to introduce price competition situations into an artificial society. We conducted a simulation experiment, from which we discovered the possibilities that cost reduction for huge-scale product recalls is efficient, and that punishment of producers that conduct no product recalls can benefit consumers. We believe this work can contribute to supporting not only government staff for improving product recall systems, but also executive officers of product companies for deliberating their strategies of recall decisions.

Keywords: Multi-agent simulation, artificial society, multi-objective optimization, evolutionary computation,genetic programming.

1. Introduction

In recent years, accidents and product recalls related to product defects have come to pose difficulties for corporations, threatening numerous industries worldwide (*e.g.*Ref. [1]). Appropriate execution of product recalls must be done to protect society. Nevertheless, the decision of whether or not to conduct product recalls is left to producers in many industries and countries [2]. Therefore, improving product recall systems necessitates assessment of decision-making by producers and their consumers related to product recalls.

Some studies of product recalls have been pursued from the viewpoint of relationships between product producers and consumers [3, 4] or from the viewpoint of economic aspects [5, 6]. However, most existing studies have adopted empirical approaches, *i.e.*, based solely on facts revealed by case studies or social surveys. It can be said that an empirical approach is no longer adequate for complex and diverse modern society because the predictive power of such approaches is limited. Therefore, discussions using a quantitative and predictive approach that can predict future events are expected.

As a quantitative and predictive approach, we have proposed and developed a fundamental social simulation model for product recall systems using a multi-agent system [7]. To reflect real-world societies

Corresponding author: Tetsuroh Watanabe, Ph.D. in Engineering, research fields: evolutionary computation, multi-agent simulation, machine learning.

and to achieve effective learning of agents' decision-making, a co-evolution model and an evolutionary computation methodology with producer agents and consumer agents were used. Results show that the proposed model was useful for predicting what might happen if various design variables of a recall system are changed.

However, the major shortcoming with the previous study was that selling prices of products sold by producer agents were completely fixed because of simplicity. In conjunction with this limitation, the model disregards the amounts of payments to consumer agents. In the real world, selling prices vary from producer to producer even in the same category. Then consumers observe selling prices when they choose products for buying. Consideration of price variation is necessary to improve the simulation accuracy.

To address this problem, the aim of this paper is to introduce a variable price model into the simulation model, and to analyze behaviors of producer and consumer agents under the price-competition situation. For this study, we simultaneously consider both consumers' satisfaction and the payment amounts using a multi-objective evolutionary algorithm (MOEA). Furthermore, we propose a new method: multi-objective likability of producer agents for evaluating the probability of being chosen by consumer agents. We analyze a distribution of agents with clustering method for avoiding arbitrariness and subjectivity in an analysis. Then we obtain suggestions for improving product recall systems in the real world.

2. Simulation Model

2.1 Assumed category of products

Producer agents sell products. Consumer agents buy and use them. In this study, no specific category of products is assumed, but it is assumed that many producers sell products of the same category and with similar specifications at various prices. Products are also assumed to cause severe accidents. Home electronics and motor vehicles are examples of this class.

2.2 Layered Co-Evolution Model

In the simulation model, Genetic Programming (GP) [8] and the co-evolution model [9] are applied together as a learning model for producers and consumers. Here, *Layered Co-Evolution Model*, an overview of which is shown in Figure1, is a unique simulation model adopted for this work.

The artificial society in the simulation environment has agents of two types: *producer agents*and*consumer agents*, as shown in Figure1. Producer agents on the upper layer have their *users* (consumer agents) on the lower layer. Each consumer agent belongs to the *user group* of a certain producer agent. These agents of two types evolve separately for their own convenience, *i.e.*, co-evolution.

Consumer agents can move to the user group of another producer agent. As described herein, we call this action *migration*. The destination producer of migration is chosen probabilistically (as described in Section 3.3). It is important for producer agents not only to retain loyal customers but also to increase the probability of being chosen in migration.

Some existing studies of real-product marketing simulations based on a multi-agent system have been described in the literature [10, 11, 12]. However, either producer agents or consumer agents can move and learn in these studies. A Layered Co-Evolution Model presents the advantage of enabling both producer agents and consumer agents to take action and to evolve in parallel.

2.3 Simulation flow

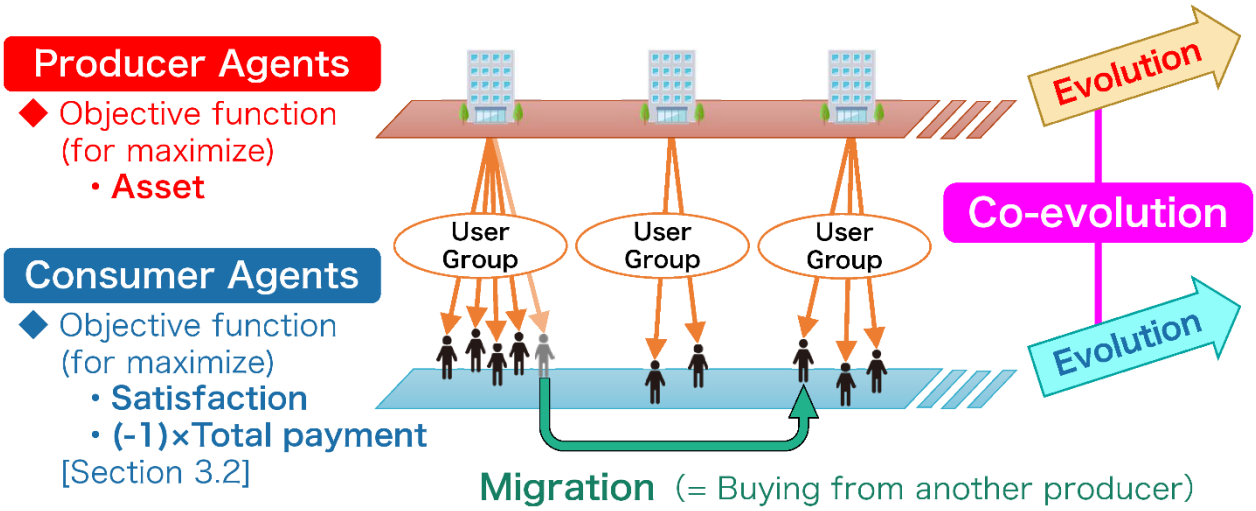


Fig. 1 Overview of Layered Co-Evolution Model. Each consumer agent belongs to the user group of a certain producer agent. Consumer agents can move to the user group of another producer agent: migration.

The simulation flow of this study is presented in Figure 2. The simulation flow comprises two parts: *Agent Optimization Flow*and *Social Simulation Flow*. Social Simulation Flow is the internal loop within Agent Optimization Flow.

Agent Optimization Flow is similar to the ordinary Genetic Algorithm (GA), promoting agents' learning process toward the direction in which agents can optimize the value of fitness function. Social Simulation Flow is the phase in which agents take actions and events occur in a time interval, called a term. We abstract factors related to discussing product recalls according to the existing studies [3, 4, 5, 6], and add actions of agents and events into Social Simulation Flow. At the end of Social Simulation Flow, the fitness values evaluated for all agents are fed back to Agent Optimization Flow.

Each producer agent has *assets*. Each consumer agent has *satisfaction* as their own parameters. The parameter values increase or decrease at the points marked $+$ and $-$ in Figure 2. Regarding satisfaction, we assume that consumers' satisfaction is obtained merely using products and assume that satisfaction of other origins is beyond the scope of this paper.

As for producer agents, assets constitute the fitness function of producer agents. It is used for Roulette Selection in the natural selection. In other words, a probability of producer agent p being selected in the natural selection $\left(P_p^{\mathrm{ProSel}}\right)$ is proportional to its assets, formulated as

$$P_p^{\mathrm{ProSel}} = \frac{a_p}{\sum_{i \in \mathcal{U}^{\mathrm{Pro}}} a_i}, \qquad (1)$$

where a_p stands for assets of p and $\mathcal{U}^{\mathrm{Pro}}$ denotes the whole population of producer agents. p's assets are evaluated as zero if p goes bankrupt. Selected producer agents are duplicated, and are put into the population of the subsequent generation.

As for consumer agents, satisfaction is a fitness function in a company with the amount of payment. It is used in natural selection (as described in Section 3.2).

In addition, each product has its lifetime ℓ. As described in this paper, ℓ is completely fixed as 12 terms. When a consumer agent uses a product through ℓ continuously, the consumer agent must buy a new product from the current producer even if the consumer agent does not migrate.

2.4 Trust and Total Trust

Each consumer agent has Trust as a parameter value. Trust represents the degree of a consumer's confidence in the producer agent whose product the consumer uses. Trust values increase, decrease, or are reset to zero at the points marked $+$, $-$, and 0 in Figure 2.

p). Total Trust represents the reputation or word-of-mouth of a producer in the real world. Greater Total Trust results in a greater probability of being chosen in migration (as described in Section 3.3).

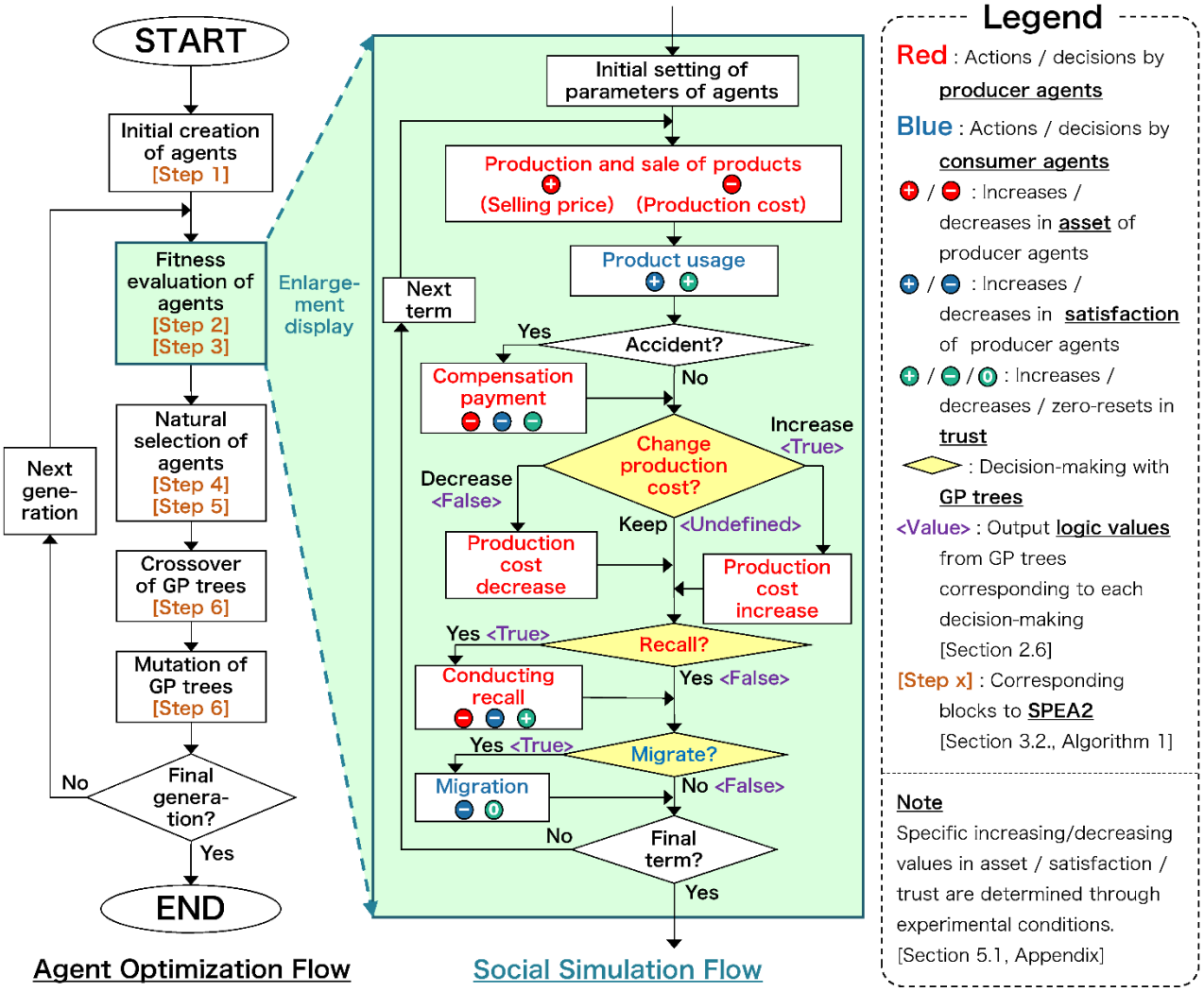


Fig. 2 Overview of Layered Co-Evolution Model. Each consumer agent belongs to the user group of a certain producer agent. Consumer agents can move to the user group of another producer agent: migration.

In this connection, each producer agent has *Total Trust* as a parameter. Total Trust is the summation of users' Trust values. Producer agent p's Total Trust ($Trust_p$) is formulated as

$$Trust_p = \sum_{c \in \mathcal{S}_p} trust_c, \tag{2}$$

where $\mathcal{S}_p$ signifies the user group of p, and $trust_c$ stands for a trust value of consumer agent c (user of

2.5 Accident probability model

The accident probability of products varies from producer agent to producer agent. Producer agent p has probability of causing a product accident (λ_p) calculated as

$$\lambda_p = \frac{\beta}{Cost_p} \cdot \gamma^{Recall_p}, \tag{3}$$

Agent type	Decision-making type	Input	Output		
		Logic value type	True	Undefined	False
Producer	Conduct Recall?	LV_2	Yes	———	No
	Change cost?	LV_3	Increase	Keep	Decrease
Consumer	Migrate?	LV_2, LV_{Event}	Yes	———	No

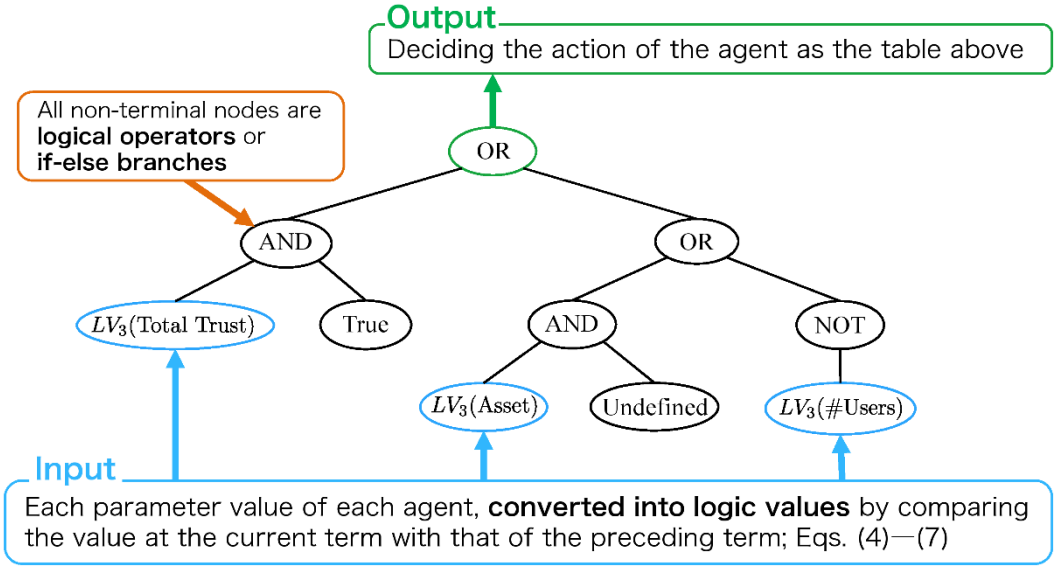


Fig. 3 Overview of Logic Value Typed GP with a GP tree example.

where β and γ are constants assigned respectively as experimental conditions $(\beta > 0,\ 0 < \gamma < 1)$, $Cost_p$ denotes the production cost of $p(Cost_p > 0)$, and $Recall_p$ represents the cumulative number of product recalls by $p(Recall_p \geq 0)$.

Producer agents can reduce their accident probability by raising their production cost or conducting product recalls, through their decision-making with the GP tree. Eq. (3) reflects the real-world situation in which producers can improve their product reliability by increasing production costs or carrying out product recalls.

2.6 Logic Value Typed GP

All agents respectively decide how they act at the yellow box in Figure 2. For agents' decision-making, we use a proposed method: *Logic Value Typed GP*: an extended method from Booleanized GP [13, 14, 15].

Agent has its own GP tree. Each parameter value of each agent is converted into logic values by comparing the value at the current term (v_{now}) with the previous term (v_{prev}) as

$$LV_2 = \begin{cases} \text{True} & (\text{if} \quad \delta > 0) \\ \text{False} & (\text{if} \quad \delta \leq 0) \end{cases} \tag{4}$$

$$LV_3 = \begin{cases} \text{True} & (\text{if} \quad \delta > dm) \\ \text{Undefined} & (\text{if} \quad |\delta| \leq dm) \\ \text{False} & (\text{if} \quad \delta < dm) \end{cases} \tag{5}$$

$$\delta = v_{\text{now}} - v_{\text{prev}} \tag{6}$$

where LV_2 is the input logic value for decision-making with two options, LV_3 is one with three options, and Undefined is "the third logic value" in the three-valued logic theory [16, 17, 18, 19, 20], and dm is *Disregard Margin*, which is for avoiding unstable fluctuations in output $(dm \geq 0)$. Disregard Margin is assigned as an experimental condition for each parameter type (*e.g.* assets).

In addition, another type of logic value LV_{Event} is used in this method, as

$$LV_{\text{Event}} = \begin{cases} \text{True} & \begin{pmatrix}\text{if the agent}\\ \text{encounters}\\ \text{an } event\end{pmatrix} \\ \text{False} & \begin{pmatrix}\text{if the agent}\\ \text{does not}\\ \text{encounters}\\ \text{an } event\end{pmatrix} \end{cases} \quad (7)$$

where $event$ is what agents possibly encounter (*e.g.* product accident, product recall). Logic values $LV_2, LV_3, LV_{\text{Event}}$ are put into the terminal nodes of GP tree. The logic values are calculated using basic logical operators (AND, OR, NOT, IF-ELSE). The output logic value, which determines the decision-making content, is finally obtained. The correspondence between decisions and the outputs from Logic Value Typed GP is described in Figure 2 with <Value> format. Figure3 additionally shows an overview and an example of Logic Value Typed GP tree and corresponding decisions of agents.

We have discovered that Logic Value Typed GP presented in an earlier report [7] has advantages over the existing GP method, which uses real number values. The Logic Value Typed GP is more stable in the evolutionary process and more efficient in terms of agents' learning processes in the simulation.

3.Optimization in the Price Competition Model

3.1 Fixed cost rate

To introduce a variable price model into the simulation model, it is assumed that the cost rate is fixed. Producer agent p can change the production cost $(Cost_p)$ following GP tree output. The selling price $(Price_p)$ will be changed also as

$$Price_p = \frac{Cost_p}{R}, \quad (8)$$

where R is the fixed cost rate assigned as an experimental condition $(0 < R < 1)$.

3.2 Multi-objective optimization on consumer agentswithMOEA

As described in this paper, the selling price of a product varies from producer to producer. It is therefore important to address not only satisfaction but also the amounts of payment of consumer agents. To optimize both satisfaction and payment simultaneously, we use a *multi-objective evolutionary algorithm (MOEA)* into Agent Optimization Flow.

To adopt MOEA, it is necessary to define the optimization directions. In the real world, it is natural to infer that larger satisfaction and less payment are better for most consumers. We accordingly assign the objective functions for maximization on consumer agents as shown below.

1. Satisfaction
2. $(-1) \times$ Total amount of payment

According to the definition of dominance relation [21], it can be described that

c_1 dominates c_2
if $Sat_{c_1} \geq Sat_{c_2}$ and $Pay_{c_1} \leq Pay_{c_2}$,

where Sat_c stands for satisfaction and Pay_c signifies the total amount of payment of consumer agent c.

We adopt *Strength Pareto Evolutionary Algorithm 2 (SPEA2)* [22] for optimization of consumer agents. SPEA2 is a leading MOEA. It can realize fine-grained fitness assignment to each. In the principle of multi-objective optimization in this simulation, it is also possible to use other MOEA, such as NSGA-II [23], MOEA/D [24], and NSGA-III [25].

The SPEA2 algorithm modified for this simulation is presented in Algorithm 1. The corresponding steps in Agent optimization flow are presented in Figure 2 with [Step x] format. In respect of SPEA2 Fitness $f^{\text{SPEA 2}}$ assigned at Step 3 in Algorithm 1, better consumer agent (having larger satisfaction or less payment) has less $f^{\text{SPEA 2}}$, calculated using Algorithm 2. At this point, Figure4 presents an example of assigning $f^{\text{SPEA 2}}$ to consumer agents. In addition, truncation operator at Step 4 in Algorithm 1 is described in

Algorithm 1 SPEA2 main flow [22] modified for this simulation.

Input: N (Population size of consumer agents)
$\overline{N}$ (Archive size of consumer agents)
Number of consumer agents in Social Simulation Flow: $\left|\mathcal{U}^{\text{Con}}\right| = N + \overline{N}$

Step 1: **Initialization:**
Generate an initial population $\mathcal{P}_0$ and create an empty archive $\overline{\mathcal{P}}_0 = \emptyset$. Set the number of evolved generation $gen = 0$.

Step 2: **Evaluation of each objective function:**
Run Social Simulation Flow with consumer agents = $\mathcal{P}_{gen} + \overline{\mathcal{P}}_{gen}$. Evaluate objective function (satisfaction and $-$payment) using parameter values at the final term.

Step 3: **SPEA2 Fitness assignment:**
Calculate and assign SPEA2 Fitness $f^{\text{SPEA 2}}$ to each consumer agent using objective function values at Step 2. Algorithm 2 gives more details related to this step.

Step 4: **Environmental selection:**
Copy all non-dominated consumer agents in $\mathcal{P}_{gen}$ and $\overline{\mathcal{P}}_{gen}$ to $\overline{\mathcal{P}}_{gen+1}$. If $\left|\overline{\mathcal{P}}_{gen+1}\right| > \overline{N}$, then reduce $\overline{\mathcal{P}}_{gen+1}$ using truncation operator. Alternatively, if $\left|\overline{\mathcal{P}}_{gen+1}\right| < \overline{N}$ then fill $\overline{\mathcal{P}}_{gen+1}$ by copying with dominated consumer agents in $\mathcal{P}_{gen}$ and $\overline{\mathcal{P}}_{gen}$ in ascending order of $f^{\text{SPEA 2}}$ (= in descending order of better consumer agents).

Step 5: **Mating selection:**
Select N consumer agents from $\overline{\mathcal{P}}_{gen+1}$ using Binary Tournament Selection (Binary Tournament Selection is Tournament Selection with tournament size = 2, *i.e.*, select two agents randomly and select the better agent). A better consumer agent (having larger satisfaction or less payment) has less $f^{\text{SPEA 2}}$.

Step 6: **Variation:**
Apply crossover and mutation operators to the agents selected at Step 5. Set $gen \leftarrow gen + 1$, and go to Step 2.

Algorithm 3, the same process with that in the original SPEA2 [22].

Table 1 presents the evolutionary computation methods for each agent type, as explained previously in Sections 2 and 3.

3.3 Multi-Objective Likability

It can be inferred that consumers refer not only to reputation but also to the selling price when buying a new product in the real world. Therefore, it is necessary to consider both Total Trust and the selling price for deciding a producer agent as a destination of consumer agent migration in this simulation model.

As described in this paper, we propose a new factor: *Multi-Objective Likability* for producer agents. The probability of being chosen as the destination of consumer's migration is determined by Roulette Selection based on Multi-Objective Likability. In other words, $P^{\text{Mig}}_{p_{\text{from}} \to p_{\text{dest}}}$, which is producer agent p_{dest}'s probability of being chosen from consumer agent c belonging to p_{from}'s user group before migration $\left(c \in \mathcal{S}_{p_{\text{from}}}\right)$, is proportional to its Multi-Objective Likability, calculated as

$$P^{\text{Mig}}_{p_{\text{from}} \to p_{\text{dest}}} = \frac{L_{p_{\text{dest}}}}{\Sigma_{i \in \mathcal{U}^{\text{Pro}} \setminus \{p_{\text{from}}\}} L_i} \quad (9)$$

$p_{\text{from}} \neq p_{\text{dest}}$,

Algorithm 2 Assigning f^{SPEA2} at Step 3 in Algorithm 1.

Input: $k = \left\lfloor \sqrt{N + \overline{N}} \right\rfloor$.

Step 3-1: Assign Pareto Strength s_c to the number of consumer agents dominated by consumer agent c.

Step 3-2: Calculate Raw Fitness $r_c = \sum_{i \in \mathcal{D}_c} s_c$ where $\mathcal{D}_c$ is the set that consists of consumer agents dominating c.

Step 3-2: Measure distances between c and all the other consumer agents. Then calculate density $d_c = \frac{1}{\sigma_c^k + 2}$, where σ_c^k is the distance between c and the k-th nearest neighbor consumer agent [26]. Distances are the Euclidean distance defined on the two-objective plane.

Step 3-3: Assign SPEA2 Fitness of consumer agent c: $f_c^{\mathrm{SPEA\,2}} = r_c + d_c$.

Algorithm 3 Truncation operator at Step 4 in Algorithm 1.

Note: The definition of distances is the same as that at Step 3-3.

Step 4-1: Search the nearest neighbor pair of consumer agents c_1 and c_2 in $\overline{\mathcal{P}}_{gen+1}$. Set $m = 2$.

Step 4-2: Calculate $\sigma_{c_1}^m$ and $\sigma_{c_2}^m$, respectively, for c_1 and c_2.

Step 4-2: Delete c_1 from $\overline{\mathcal{P}}_{gen+1}$ if $\sigma_{c_1}^m < \sigma_{c_2}^m$. Delete c_2 from $\overline{\mathcal{P}}_{gen+1}$ if $\sigma_{c_1}^m > \sigma_{c_2}^m$. Otherwise if $\sigma_{c_1}^m = \sigma_{c_2}^m$, set $m \leftarrow m + 1$, and go to Step 4-2.

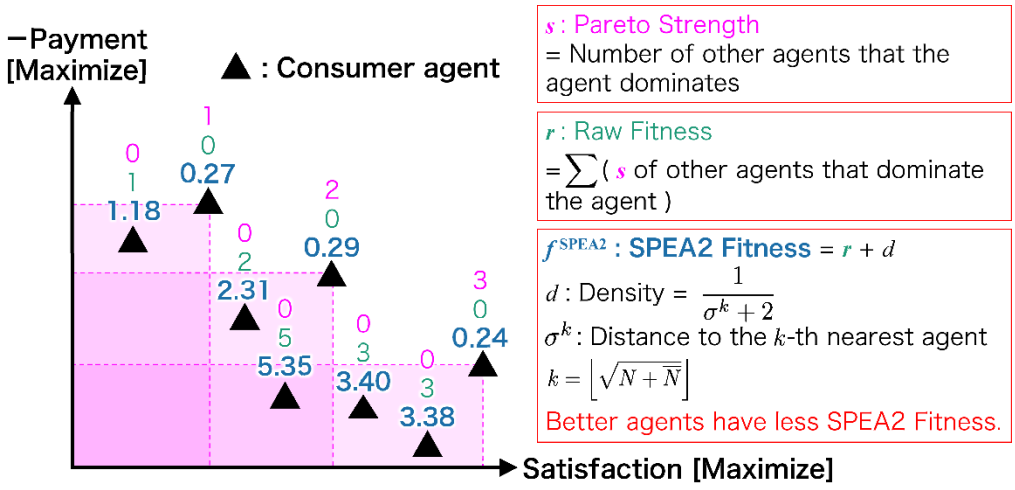


Fig. 4 Example of assigning f^{SPEA2} to consumer agents.

where L_p is the Multi-Objective Likability of p.

Next, a detailed definition of Multi-Objective Likability is given. At this point, we introduce a similar method to the algorithm of assigning Raw Fitness r_c in SPEA2 (Steps 3-1, 3-2). In this method, we deal with producer agents' two parameters:

1. Total Trust
2. (-1) ×Selling price

like two objective functions of consumer agents. At this point, we define the concept of *pseudo-domination* as follows.

p_1 pseudo-dominates p_2
if $Trust_{p_1} \geq Trust_{p_2}$ and $Price_{p_1} \leq Price_{p_2}$

Table 1 Summary of evolutionary computation methods

Agent type	Objective function (maximization)	Method for the natural selection
Producer	1. Assets	Roulette Selection based on assets
Consumer	1. Satisfaction 2. $(-1) \times$ Payment	SPEA2(Environmental selection, Mating selection)

Algorithm 4 Assigning Multi-Objective Likability

Step 1: Assign Pseudo Pareto Strength $\tilde{s}_p$ to the number of producer agents pseudo-dominated by producer agent p.

Step 2: Calculate Pseudo Raw Fitness $\tilde{r}_p = \sum_{i \in \tilde{\mathcal{D}}_p} \tilde{s}_p$ where $\tilde{\mathcal{D}}_p$ is the set that consists of producer agents pseudo-dominating p.

Step 3: Assign Multi-Objective Likability of producer agent p: $L_p = \frac{1}{\tilde{r}_p + 1}$.

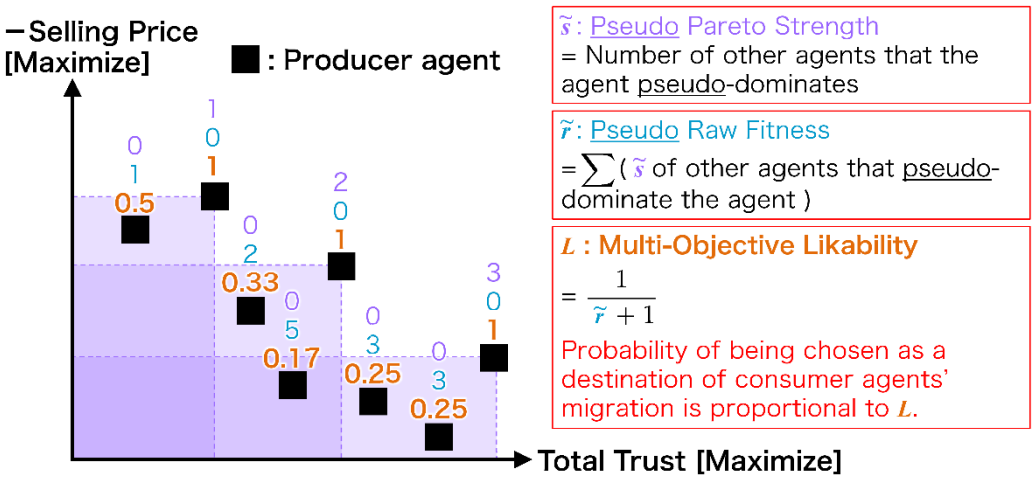


Fig. 5 Assigning Multi-Objective Likability to producer agents.

Multi-Objective Likability is assigned using this pseudo-domination concept as Algorithm 4. Additionally, Figure5 presents an example of assigning Multi-Objective Likability to producer agents.

In this model, greater Total Trust and a lower selling price lead to higher Multi-Objective Likability for the producer agent. This model can be appropriate because it can be considered that, in the real world, many consumers tend to prefer products having a better reputation or a lower price.

4.Cluster Analysis for Agents

4.1 Meaning of cluster analysis

In a real-world society, there are various tendencies of strategy. When improvement of social systems is planned, it is better to classify producers or consumers having a similar tendency to the same category.

Considering this aspect, we introduce a *clustering* method for the simulation, and analyze distributions of agents while particularly addressing the emergence of clusters. Additionally, we track product flows from

Algorithm 5 Cluster analysis for producer agents

Input: k^{Pro} (Number of clusters for producer agents)

Step 1: Apply clustering to producer agents at the final term in the final generation to clusters $\mathcal{C}_i^{\mathrm{Pro}}$ $(i = 1,2,\cdots,k^{\mathrm{Pro}})$

Step 2: Illustrate each cluster of producer agents $\mathcal{C}_i^{\mathrm{Pro}}$, and interpret meanings of each $\mathcal{C}_i^{\mathrm{Pro}}$.

Algorithm 6 Cluster analysis for consumer agents and tracking product flows

Input: k^{Con} (Number of clusters for consumer agents)

Step 1: Apply clustering to consumer agents at the final term in the final generation to clusters $\mathcal{C}_i^{\mathrm{Con}}$ $(i = 1,2,\cdots,k^{\mathrm{Con}})$.

Step 2: Illustrate each cluster of consumer agents $\mathcal{C}_i^{\mathrm{Con}}$, and interpret meanings of each $\mathcal{C}_i^{\mathrm{Con}}$.

Step 3: Track product flows, and count products based separately on those bought by which $\mathcal{C}_i^{\mathrm{Con}}$ consumer agent.

Step 4: Categorize producer agents according to whether the number of sales is ranked in M top or not, for each $\mathcal{C}_i^{\mathrm{Con}}$ counted at Step 3.

producer agents to consumer agents, and analyze inter-dependencies leading to more purchases between both agents of various types.

4.2 Methods of clustering agents and tracking products

As described in this paper, we use *K-means method* [27] for clustering agents. K-means method is widely used. Its usefulness is confirmed [28]. Clustering agents and tracking product flows are conducted as Algorithms 5 and 6.

The numbers of clusters, k^{Pro} and k^{Con}, are determined through *Elbow Method* [29, 30, 31]. In Elbow Method, clustering is processed with various numbers of clusters. The sum of squared errors (SSE) in each cluster is calculated with each number of clusters. SSE is the sum of distances from the centroid of the cluster to each agent in the cluster. SSEs are shown (horizontal axis, number of clusters; vertical axis, SSE). The best number of clusters is detected as an elbow-shaped point, *i.e.*, the convex downward point. k^{Pro} and k^{Con} can take different values from each other. However, to maintain symmetric analysis, using the same numbers of clusters for both producer agents and consumer agents is desirable.

5. Simulation Experiment

To confirm the usefulness of our proposed method, and to analyze behaviors of producer agents and consumer agents under a price-competition situation, we conducted a simulation experiment.

The simulation was conducted ten times, from which almost identical tendencies were observed. One set of results is given here because of space limitations but referring to another one does not change the following discussions or the conclusions obtained from this study.

5.1 Experimental conditions

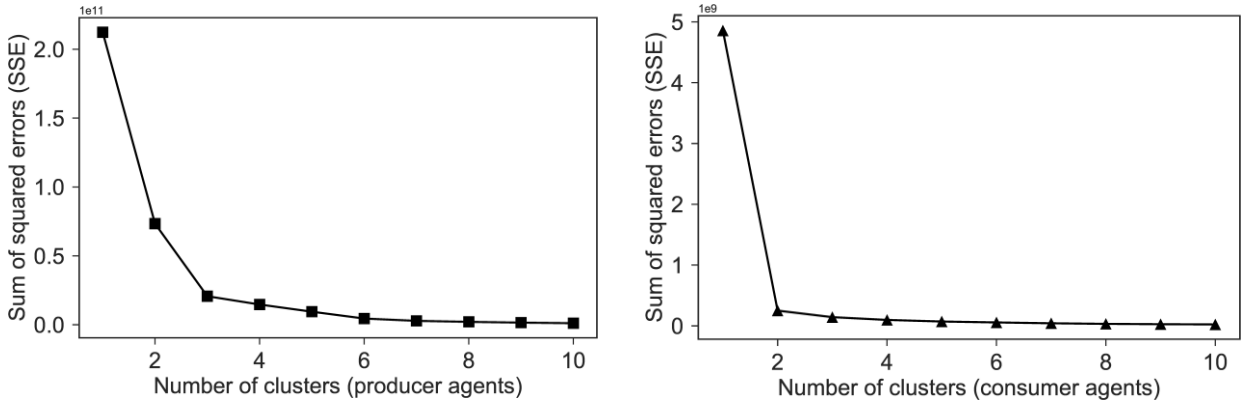
Table 2 presents main conditions of the simulation experiment. Table 4 and Table 5 in Appendix A present additional details related to experimental conditions. In addition, Table 3 presents feature vectors of each type of agent for clustering (K-means method). All feature vectors in Table 3 are values at the final term in the final generation of the simulation. With respect to M value in Algorithm 6, we set $M = 20$, which is the top 10% ranked producer agents in the number of product sales.

Table 2 Main conditions of the simulation experiment

Condition type	Value
Number of generations	300
Number of terms in each generation	120
Number of producer agents	200
Number of consumer agents	10,000
R: Fixed cost rate in Eq. (8)	0.8
$\overline{N}$: Archive size of consumer agents in Algorithm 1	5,000

Table 3 Feature vectors for clustering (K-means method)

Agent type	Dimensions of feature vectors	Number of dimensions
Producer	Assets, Total Trust, Selling price, Number of users, Number of product recalls, Number of product accidents, Number of being chosen as a destination of a migration	7
Consumer	Satisfaction, Total amount of payment, Number of migrations, Number of encountered product recalls, Number of encountered product accidents, Mean of payment at each purchase, Number of products used through a whole lifetime	7



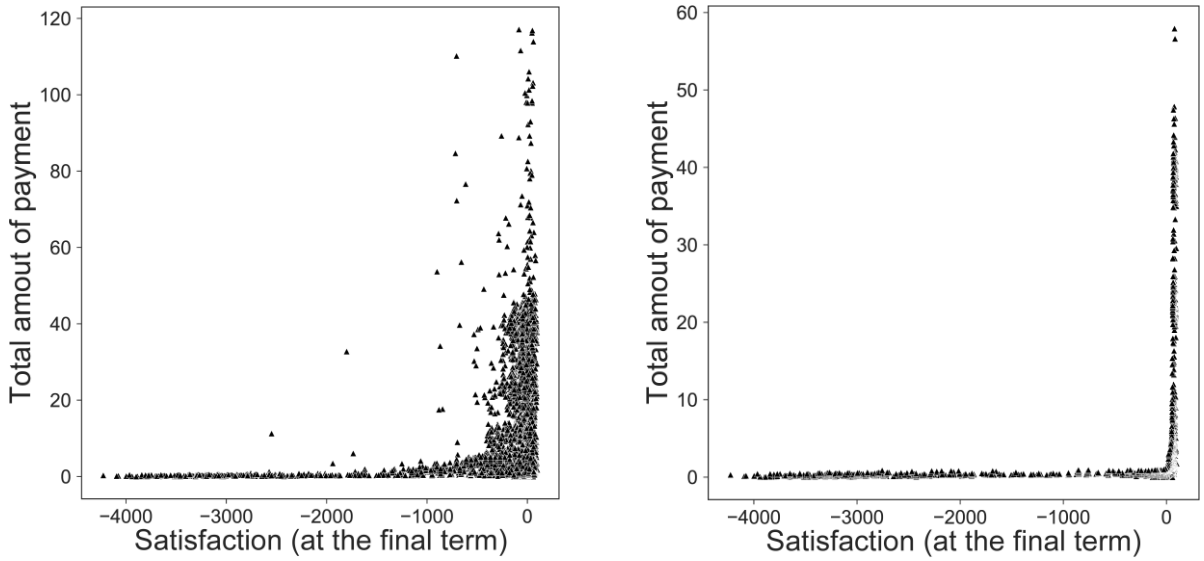
(a) Elbow Method on producer agents. (b) Elbow Method on consumer agents.

Fig. 6 Transition of SSE values at each number of clusters (Elbow Method). A clear convex downward point is apparent where the number of clusters is three on producer agents and two on consumer agents.

5.2 Determining the numbers of clusters using ElbowMethod

Before the main part of cluster analysis, we apply Elbow Method to agent distributions at the final term in the final generation. Figure6 presents results of Elbow Method on producer agents and on consumer agents. According to Figure6a, it is readily apparent that the best number of clusters for producer agents is 3 because a clear convex downward point exists. By contrast, Figure6b clarifies that the one for consumer agents is 2.

As described in Section 4.2, the same number of clusters for both agent types is preferred. Additionally,



(a) Before SPEA2 operation (consumer agents). (b) After SPEA2 operation (consumer agents).

Fig. 7 Comparing the consumer agent distribution before and after SPEA2 operation (at the 35th generation, satisfaction in x-axis vs. the total amount of payment in y-axis). Pareto Front boundary is generated by SPEA2 operation.

some concern exists that setting the number of clusters to 2 might make the analysis too simple. We therefore set $k^{\mathrm{Pro}} = k^{\mathrm{Con}} = 3$ for the cluster analysis.

5.3 Results and discussion

5.3.1 Effect of MOEA

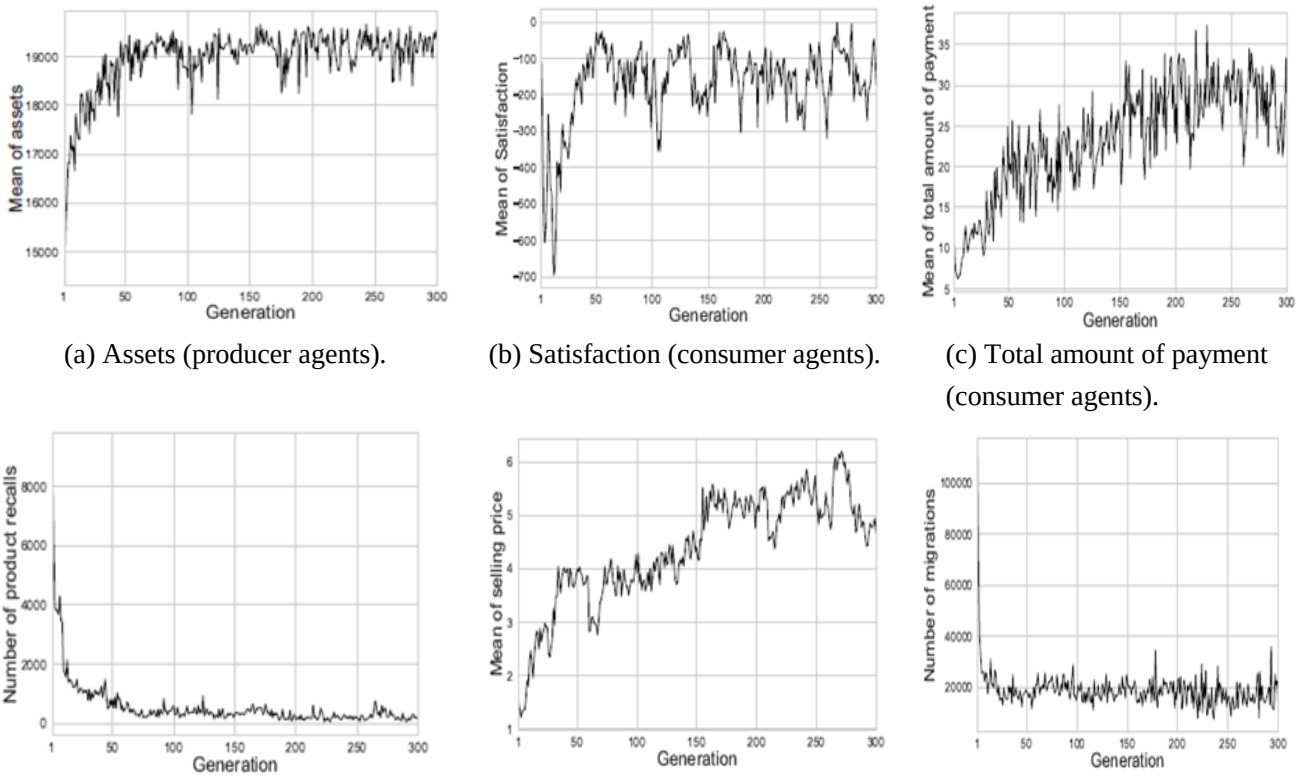
We checked whether SPEA2 method works effectively or not. Figure7 presents an example of consumer agent distribution before (Figure7a) and after (Figure7b) SPEA2 operation (satisfaction in x-axis vs. total amount of payment in y-axis). In comparison between Figure7a and Figure7b, a Pareto Front boundary is generated toward the direction of maximizing satisfaction and minimizing payment.

This is an example at 35th generation, but we observed the same phenomenon for all other generations. As expected, these results indicate that SPEA2 works effectively as a method for multi-objective optimization.

5.3.2 Evolutionary convergence

To ascertain whether the evolution converged sufficiently or not, we observed the transition of the evolution through all 300 generations. Figure8 shows evolutionary transitions at the means of parameters at the final term of each generation. The horizontal axis shows generations. The vertical axis shows the means of parameters. Figure8 presents assets of producer agents (Figure8a), with satisfaction of consumer agents (Figure8b), the total payment of consumer agents (Figure8c), the number of product recalls of producer agents (Figure8d), the selling price of producer agents (Figure8e), and the number of migrations of consumer agents (Figure8f).

This simulation includes some probabilistic factors, as shown in Figure 2. Evolutionary transitions fluctuate to some extent, even in later generations. However, evolution of agents' decision-making is stable, especially on product recall and migrations. As explained earlier, the co-evolution mechanism directly affects decision-making of agents through GP trees. It therefore can be inferred that the evolution is



(a) Assets (producer agents). (b) Satisfaction (consumer agents). (c) Total amount of payment (consumer agents).

(d) Number of product recalls (producer agents). (e) Selling price (producer agents). (f) Number of migrations (consumer agents).

Fig. 8 Transition of evolutionary process in means of parameter values of respective agents at the final terms of

converged sufficiently at the 300th generation. Taking this evolutionary behavior into account, it can be inferred that the explained in the following sections are of well-learned agents.

5.3.3 Cluster analysis of producer agents

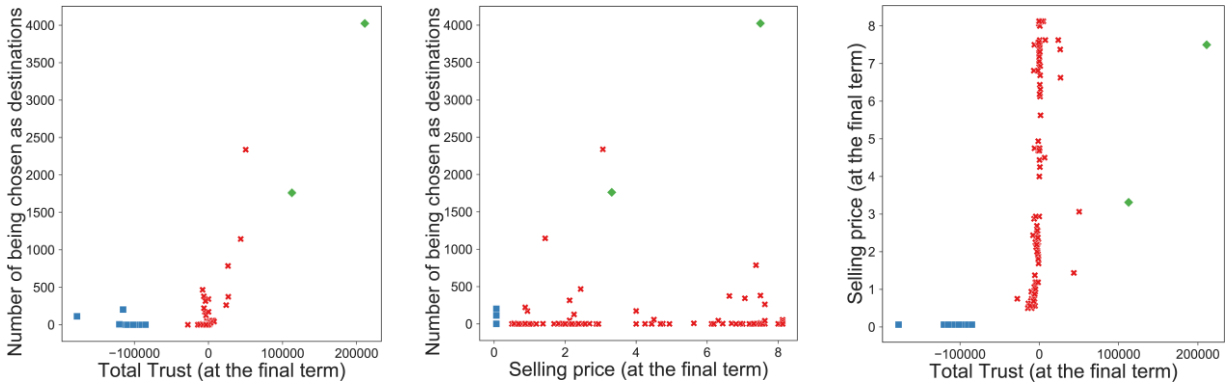
With the following Algorithm 5, we analyzed the distribution of producer agents using clustering technique. We set $k^{\mathrm{Pro}} = 3$ in Section 5.2. The producer agents are classified into three clusters using K-means method: $\mathcal{C}_1^{\mathrm{Pro}}$, $\mathcal{C}_2^{\mathrm{Pro}}$, and $\mathcal{C}_3^{\mathrm{Pro}}$.

In this section, we first verified the effectiveness of our proposed method: Multi-Objective Likability. Secondly, we investigate characteristics of respective clusters by observing illustrated clusters and results of tracking product flows from producer agents to consumer agents.

(a) Effect of Multi-Objective Likability

We investigate how Multi-Objective Likability affects agent behaviors. Figure9 portrays the distribution of producer agents with axis combinations of three types for checking the effect of Multi-Objective Likability.

Figure9a shows that a larger Total Trust engenders more times of being chosen as a destination in migrations, and the correlation coefficient $r = 0.538$. Compared with this result, Figure9b presents that a clear correlation between the selling price and the number of being chosen as destination does not exist, and that $r = 0.013$. On this point, it is inferred from Figure9c that the distribution of Total Trust is not uniform, and that the distribution provides differences



(a) Total Trust / Number chosen as destinations of migration (producer agents) [$r = 0.538$].

(b) Selling price / Number chosen as destinations of migration (producer agents) [$r = 0.013$].

(c) Total Trust / selling price (producer agents).

Fig. 9 Distribution of producer agents for investigating the effect of Multi-Objective Likability: (a) has a higher correlation coefficient r than (b) because the distribution of Total Trust has larger bias than that of selling price in (c).

of Multi-Objective Likability's effects among producer agents. However, the probability is that the selling price does not clearly affect Multi-Objective Likability because the selling price values vary in a wide range with only a little bias, $\mathcal{C}_1^{\mathrm{Pro}}$ in particular. These results suggest that future work should address the definition of Multi-Objective Likability must be more refined for reflecting little-biased parameter values.

(b) Characteristics of respective clusters

Next, we interpret the meanings of the respective clusters of producer agents, and investigate the cluster characteristics. Figure10 presents results of cluster analysis for producer agents. We interpret the three clusters of $\mathcal{C}_1^{\mathrm{Pro}}, \mathcal{C}_2^{\mathrm{Pro}}, \mathcal{C}_3^{\mathrm{Pro}}$, as follows:

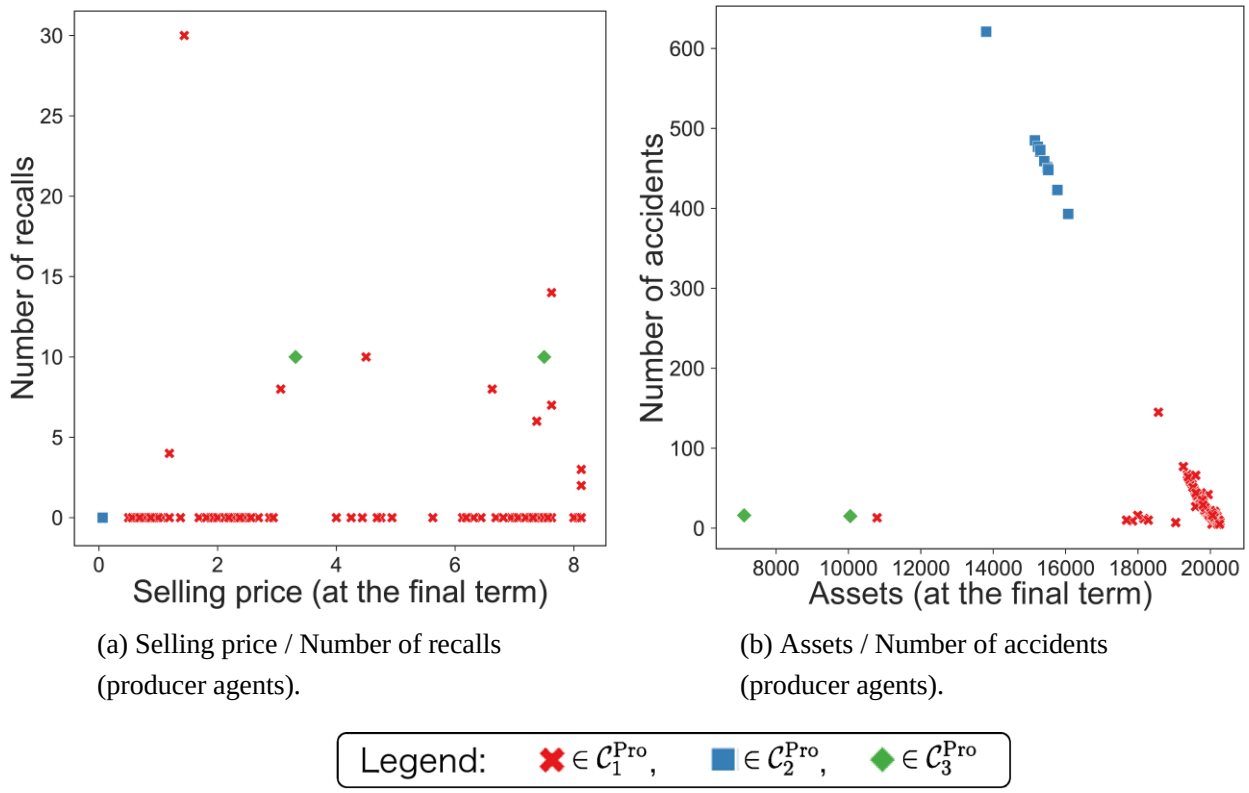
$\mathcal{C}_1^{\mathrm{Pro}}$: Ordinary producer agents vary the selling price in a wide range. In other words, producer agents outside of $\mathcal{C}_2^{\mathrm{Pro}}$ and $\mathcal{C}_3^{\mathrm{Pro}}$.

$\mathcal{C}_2^{\mathrm{Pro}}$:Undesirable producer agents: producing only low-price and poor-quality products, conducting no product recalls, and causing many accidents.

$\mathcal{C}_3^{\mathrm{Pro}}$:Desirable producer agents: carrying out product recalls honestly and restraining accidents.

According to the interpretation above, $\mathcal{C}_2^{\mathrm{Pro}}$ are unfavorable producer agents for the society. They should be corrected by improving social systems. By contrast, $\mathcal{C}_3^{\mathrm{Pro}}$ are desirable producers. In other words, they are paragons as producer agents.

$\mathcal{C}_3^{\mathrm{Pro}}$ includes a few producer agents, but it is noteworthy that $\mathcal{C}_3^{\mathrm{Pro}}$ producer agents have finally less assets than $\mathcal{C}_2^{\mathrm{Pro}}$, in spite of their sincerity. In this respect, we observed the detailed raw logs of the simulation, and discovered that huge outlays for product recalls are necessary for $\mathcal{C}_3^{\mathrm{Pro}}$ producer agents because they have numerous users based on their sincerity. This situation, in which honest product recalls cause diseconomy, is undesirable in a real-world society. It can be said that $\mathcal{C}_3^{\mathrm{Pro}}$ producer agents should therefore be helped by improving product recall systems. With respect to this point, this result suggests that it is better to improve product recall systems to realize cost reductions for huge-scale product recalls. This suggestion is the contribution of



(a) Selling price / Number of recalls (producer agents).

(b) Assets / Number of accidents (producer agents).

Fig. 10 Distribution of producer agents. Plotting markers are changed according to the cluster to which they belong.

this paper because existing studies have not described this point using quantitative or predictive approaches.

5.3.4 Cluster analysis of consumer agents

Finally, we explore cluster analysis of consumer agents. Following Algorithm 6, we observed the distribution of producer agents using clustering method. Additionally, we tracked product flows and counted the amount of sales of each producer to each consumer cluster. We set $k^{\mathrm{Con}} = 3$ in Section 5.2. Consumer agents are classified into three clusters using K-means method: $\mathcal{C}_1^{\mathrm{Con}}$, $\mathcal{C}_2^{\mathrm{Con}}$, and $\mathcal{C}_3^{\mathrm{Con}}$.

Figure11 presents results of cluster analysis for consumer agents, and also the result of tracking product sales from producer to consumer clusters. We interpret the three clusters shown in Figure11a, $\mathcal{C}_1^{\mathrm{Con}}$, $\mathcal{C}_2^{\mathrm{Con}}$, $\mathcal{C}_3^{\mathrm{Con}}$, as follows:

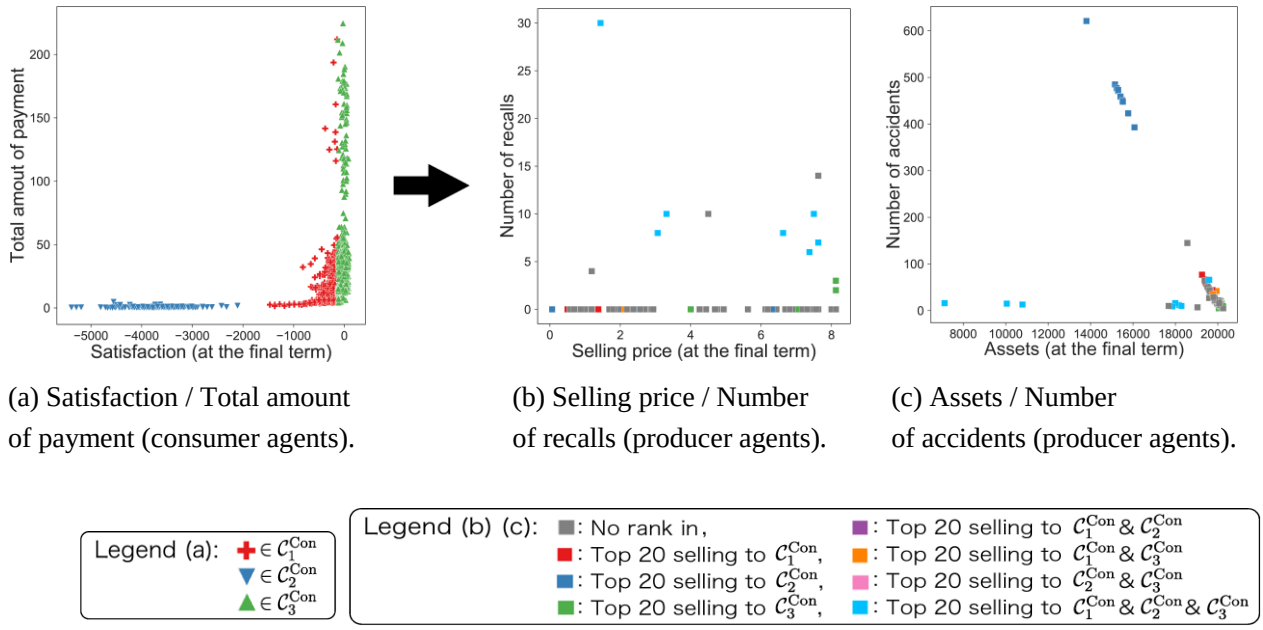
$\mathcal{C}_1^{\mathrm{Con}}$:Ordinary consumer agents: accepting both large payment amounts and less satisfaction to some degree. In other words, agents outside of $\mathcal{C}_2^{\mathrm{Con}}$ and $\mathcal{C}_3^{\mathrm{Con}}$.

$\mathcal{C}_2^{\mathrm{Con}}$:Stingy consumer agents: paying little money and disregarding satisfaction, even if they themselves encounter accidents.

$\mathcal{C}_3^{\mathrm{Con}}$:Satisfaction-preferring consumer agents: obtaining high satisfaction and disregarding their amounts of payment.

Moreover, Figure11b and Figure11c present distributions of producer agents that are color-coded according to whether or not the number of sales is in the top 20.

Based on the results, we would like to emphasize that producer agents in 20 top sales to only $\mathcal{C}_2^{\mathrm{Con}}$ (plotted with dark-blue square marker in Figure11b and Figure11c) are similar to $\mathcal{C}_2^{\mathrm{Pro}}$ in Figure10. We described in Section 5.3.3 that $\mathcal{C}_2^{\mathrm{Pro}}$ are undesirable producer agents for society because of a lack of product recalls and frequently occurring accidents. These



(a) Satisfaction / Total amount of payment (consumer agents).

(b) Selling price / Number of recalls (producer agents).

(c) Assets / Number of accidents (producer agents).

Fig. 11 Distribution of consumer agents, whose plotting markers differ according to their cluster, and distribution of producer agents color-coded according to whether the number of sales is ranked in or not.

results illustrate that $\mathcal{C}_2^{\mathrm{Con}}$ are capital sources of $\mathcal{C}_2^{\mathrm{Pro}}$. In other words, undesirable producer agents can avoid bankruptcy by virtue of stingy consumer agents despite their stingy risk-taking attitude.

In this respect, the possibility exists that punishment of undesirable producers also means saving economization-preferring consumers from product accidents, such as "killing two birds with one stone." This possibility can be regarded as natural from a qualitative perspective. However, a contribution of this paper is this suggestion with a qualitative and predictive perspective.

In addition, with respect to $\mathcal{C}_3^{\mathrm{Con}}$, this cluster includes satisfaction-preferring consumer agents. It can be said that the consumer agents in $\mathcal{C}_3^{\mathrm{Con}}$ are not stingy and that they pay a greater amount of money for maximizing their satisfaction. They prefer producer agents dealing with expensive products. In fact, according to Figure11b and Figure11c, producer agents in 20 top sales to only $\mathcal{C}_3^{\mathrm{Con}}$ (plotted with green square marker) sell expensive products.

6. Conclusions

In summary, we have constructed a social simulation model for analyzing product recall systems in a setting with price competition. As a result of the simulation experiment, we have discovered the possibility that cost reduction for huge scale product recalls is efficient, and that punishment to producers conducting no product recall leads to benefits for consumers.

As described in Section 2.2, most existing marketing studies using multi-agent simulation have only tackled producers' actions. As described in this paper, we accomplish the simulation experiment that incorporates not only producers' actions but also consumers' behaviors, including decision-making for product recalls. Discovering the suggestions using quantitative and predictive approaches is the contribution of this paper.

Regarding limitations of this research, some problems remain to be resolved. A noteworthy issue is difficulties in analyzing micro-level behavior of the

agents, *i.e.*, interpretation of GP trees in particular. Analyzing details of GP tree elements is difficult because of its complexity, as typified by the *bloat phenomenon* [32]. Despite the limitations, however, this study presents a new approach for investigating means of improving product recall systems considering the decision-making and adaptation processes of both producers and consumers from the viewpoint of co-evolution.

We believe that this work can contribute to support not only of government staff for improving product recall systems, but also of executive officers of product companies for deliberating their strategies of recall decisions. As a subject of future work, micro-level analysis is necessary for additional validation of the results obtained from the proposed model.

References

[1] Reports, C. 2016. "Takata Airbag Recall—Everything You Need to Know—What This Recall Means to You and What Actions You Should Take." http://www.consumerreports.org/cro/news/2016/05/everything-you-need-to-know-about-the-takata-air-bag-recall/index.htm.

[2] Abbott, H. 1991.*Managing Product Recall*. Pitman Publishing.

[3] Berman, B. 1999. "Planning for the Inevitable Product Recall." *Business Horizons* 42 (2): 69-78.

[4] Laufer, D., and Jung, J. M. 2010. "Incorporating Regulatory Focus Theory in Product Recall Communications to Increase Compliance with a Product Recall." *Public Relations Review* 36 (2): 147-51.

[5] Davidson, W. N., and Worrell, D. L. 1992. "Research Notes and Communications: The Effect of Product Recall Announcements on Shareholder Wealth." *Strategic Management Journal* 13 (6): 467-73.

[6] Chen, Y., Ganesan, S., and Liu, Y. 2009. "Does a Firm's Product-Recall Strategy Affect Its Financial Value? An Examination of Strategic Alternatives during Product-Harm Crises." *Journal of Marketing* 73 (6): 214-26.

[7] Watanabe, T.,Kanno, T., and Furuta, K. 2017. "Social Simulation for Improving Product Recall Systems by Layered Co-evolution Model and Logic Value Typed Genetic Programming (in Japanese)." *Transaction of the Japanese Society for Evolutionary Computation* 8 (2): 36-51.

[8] Koza, J. R. 1990.*Genetic Programming: A Paradigm for Genetically Breeding Populations of Computer Programs to Solve Problems*. Dept. of Computer Science, Stanford Univ., no. STAN-CS-90-1314.

[9] Hillis, W. D. 1990. "Co-evolving Parasites Improve Simulated Evolution as an Optimization Procedure." *Physica D: Nonlinear Phenomena* 42: 228-34.

[10] Klos, T. B., and Nooteboom, B. 2001. "Agent-Based Computational Transaction Cost Economics." *Journal of Economic Dynamics and Control*.

[11] Baiman, S., Fischer, P. E., and Rajan, M. V. 2001. "Performance Measurement and Design in Supply Chains." *Management Science*.

[12] Mizuno, M., and Nishiyama, N. 2003. "Interacting TV Viewers: A Case of Empirical Agent-Based Modeling and Simulation for Business Applications." *Advances in Complex Systems* 6 (3): 361-73.

[13] Eggermont, J., Eiben, A., and van Hemert, J. 1999. "A Comparison of Genetic Programming Variants for Data Classification." In *Proceedings of 3rd International Symposium on Advances in Intelligent Data Analysis (IDA 99)*, 281-90.

[14] Bojarczuka, C. C.,Lopesa, H. S., Freitasb, A. A., and Michalkiewicz, E. L. 2004. "A Constrained-Syntax Genetic Programming System for Discovering Classification Rules: Application to Medical Data Sets." *Artificial Intelligence in Medicine* 30 (1): 27-48.

[15] Sakprasat, S., and Sinclair, M. C. 2007. "Classification Rule Mining for Automatic Credit Approval Using Genetic Programming." In *Proceedings of 2007 IEEE Congress on Evolutionary Computation (CEC 2007)*, 548-55.

[16] Łukasiewicz, J. 1920. "On Three-Valued Logic." *Ruch Filozoficzny* 5: 170-1.

[17] Kleene, S. C. 1938. "On a Notation for Ordinal Numbers." *Journal of Symbolic Logic*3: 150-5.

[18] Kleene, S. C. 1952.*Introduction to Metamathematics*. North-Holland Publishing Co.

[19] Sobociński, B. 1952. "Axiomatization of a Partial System of Three-Valued Calculus of Propositions." *Journal of Computing Systems* 1: 23-55.

[20] Ciucci, D., and Dubois, D. 2013. "A Map of Dependencies among Three-Valued Logics." *Information Sciences* 250: 162-77.

[21] Deb, K. 2012.*Innovization: Innovative Solution Principles Using Multiobjective Optimization*. Springer-Verlag.

[22] Zitzler, E.,Laumanns, M., and Thiele, L. 2001. "SPEA2: Improving the Performance of the Strength Pareto Evolutionary Algorithm." *Technical Report* 103.

[23] Deb, K., Agrawal, S.,Pratab, A., and Meyarivan, T. 2002. "A Fast Elitist Non-dominated Sorting Genetic Algorithm for Multi-objective Optimization: NSGA-II." *IEEE*

Transaction on Evolutionary Computation 2 (6): 182-97.

[24] Zhang, Q., and Li, H. 2007. "MOEA/D: A MultiobjectiveEvolutionary Algorithm Based on Decomposition." *IEEE Transactions on Evolutionary Computation* 11 (6): 712-31.

[25] Deb, K., and Jain, H. 2014. "An Evolutionary Many-Objective Optimization Algorithm Using Reference-Point-Based Nondominated Sorting Approach, Part I: Solving Problems with Box Constraints." *IEEE Transaction on Evolutionary Computation* 18 (4): 577-601.

[26] Silverman, B. W. 1986. "Density Estimation for Statistics and Data Analysis." In *London: Chapman and Hall*.

[27] MacQueen, J. 1967. "Some Methods for Classification and Analysis of Multivariate Observations." In *Proceedings of 5th Berkeley Symposium on Mathematical Statistics and Probability*, vol. 1. University of California Press, 281-97.

[28] Xu, R., and II, D. C. W. 2005. "Survey of Clustering Algorithms." *IEEE Transactions on Neural Networks*16 (3): 645–78.

[29] Thorndike, R. L. 1953. "Who Belongs in the Family?" *Psychometrika* 18 (4): 267-76.

[30] Ketchen, D. J., and Shook, C. L. 1996. "The Application of Cluster Analysis in Strategic Management Research: An Analysis and Critique." *Strategic Management Journal* 17(60): 441-58.

[31] Goutte, C.,Toft, P.,Rostrup, E., Nielsen, F. Å ., and Hansen, L. K. 1999. "On Clustering fMRI Time Series." *NeuroImage* 9 (3): 298-310.

[32] Nordin, P.,Francone, F., and Banzhaf, W. 1995. "Explicitly Defined Introns and Destructive Crossover in Genetic Programming." In *Proceedings of the Workshop on Genetic Programming: From Theory to Real-World Applications*, 6-22.

Appendix

A. Detailed Experimental Condition

Table 4 presents detailed conditions of the simulation experiment. Table 5 presents parameters selected as candidate terminal nodes of GP trees. Table 4 and Table 5 are complements of the main conditions described in Table 2 in Section 5.1.

Table 4 Detailed conditions of the simulation experiment.

Condition type	Value
ℓ: Lifetime of product	12 terms
Initial assets of a producer agent	20,000
Initial satisfaction of a consumer agent	0
Initial trust of a consumer agent	0
Initial production cost of a producer agent	0.5
Minimum production cost	0.05
Changing unit of production cost	0.05
Additional cost for a product recall per product	1
Satisfaction when a consumer agent uses a product normally	$+1$
Trust when a consumer agent uses a product normally	$+1$
Compensation cost when a producer agent causes a product accident	10
Satisfaction when a consumer agent encounters a product accident independently	-100
Trust when a consumer agent encounters a product accident independently	-100
Satisfaction when another consumer agent belonging to the same user group encounters a product accident	-5
Trust when another consumer agent belonging to the same user group encounters a product accident	-5
Satisfaction when a consumer agent encounters a product recall	-1
Trust when a consumer agent encounters a product recall	$+10$
Constant value β in Eq. (3): base value of product accident probability	7.5×10^{-3}
Constant value γ in Eq. (3): reducing rate of product accident probability by a product recall	0.5
Disregard Margin dm (assets)	2
Disregard Margin dm (number of accidents)	0
Disregard Margin dm (number of users)	2
Disregard Margin dm (total trust)	55
Disregard Margin dm (Z score of selling price)	1.0×10^{-3}
Range of depth of initial GP trees	[0, 3]
Range of depth of GP trees through the simulation flow	[0, 3]
Crossover rate	0.8
Crossover method	one point
Mutation rate	0.02
Mutation method	one point
Range of depth of GP trees generated by mutation operation	[0, 2]

Table 5 Parameters selected as candidate terminal nodes of GP trees.

Agent type	Parameters selected as candidate terminal nodes ($LV_2, LV_3, LV_{\text{Event}}$)
Producer	Assets
	Total Trust
	Number of users
	Number of product accidents
	Z score of selling price
Consumer	Satisfaction
	Trust
	Z score of selling price of the producer agents whose user group a consumer belongs to
	Whether a consumer encounters an accident during the term
	Whether other user in the same user group encounters an accident in the term
	Whether a user encounters a product recall