Using Evolving Agents to Critique Subjective Music Compositions

Chuen-Tsai Sun, Ji-Lung Hsieh
Department of Computer Science
National Chiao Tung University
1001 Ta Hsueh Road, Hsinchu, Taiwan, China
{ctsun, gis91572}@cis.nctu.edu.tw

Chung-Yuan Huang
Department of Computer Science and Information Engineering
Chang Gung University
259 Wen Hwa 1st Road, Taoyuan, Taiwan, China

Abstract

The authors describe a recommender model that uses intermediate agents to evaluate a large body of subjective data according to a set of rules and make recommendations to users. After scoring recommended items, agents adapt their own selection rules via interactive evolutionary computing to fit user tastes, even when user preferences undergo a rapid change. The model can be applied to such tasks as critiquing large numbers of music or written compositions. In this paper we use musical selections to illustrate how agents make recommendations and report the results of several experiments designed to test the model's ability to adapt to rapidly changing conditions yet still make appropriate decisions and recommendations.

1. Introduction

Since the birth of the Netscape web browser in 1994, millions of Internet surfers have spent countless hours searching for current news, research data, and entertainment—especially music. Users of Apple's Musicstore can choose from 2,000,000 songs for downloading. Having to deal with so many choices can feel like a daunting task to Internet users, who could benefit from efficient recommender systems that filter out low-interest items [1-3].

Some of the most popular Internet services present statistical data to point users to items that they might be interested in. News websites place stories that attract the broadest interest on their main pages, and commercial product stores such as amazon.com use billboards to list current book sales figures and to make recommendations that match collected data on user behaviors. However, these statistical methods are less useful for making music, image, or other artistic product recommendations to users whose subjective preferences can cross many genres. Music selections are often made based on mood or time of day [4, 5].

Two classical approaches to personalized recommender systems are content-based filtering and collaborative filtering. Content-based filtering methods focus on item content analyses and recommend items similar to interested items given by user in the past [1, 6], while the experts use collaborate filtering method to make the group of users with common interests share their accessed information [7-9]. Common design challenges of previous approaches include:

1. When the recommended item is far different from the user's preferences, the user still can only access or select these system-recommended items, and cannot access the potential good items which never appear in the set of recommended items. This problem can be solved possibly with an appropriate feedback mechanism [7].
2. In a collaborative filtering approach, new items may not be selected due to sparse rating histories.
3. User preferences may change over time or according to the moment, situation, or mood [4, 5].
4. Because of the large body of subjective compositions, the required large amount of time for forming suitable recommendations needs to should be reduced [4, 5].

In light of these challenges, we have created a music recommender system model which was designed to reduce agent training time through user feedback. Model design consists of three steps: a) content-based filtering methods are used to extract item features, b) a group of agents make item recommendations, and c) an evolution mechanism is used to make adjustments according to the subjective emotions and changing tastes of users.

2. Related Research

2.1. Recommender Systems

The two major components of recommender systems are items and users. Many current systems use algorithms to make recommendations regarding music

1-4244-0605-6/06/$20.00 ©2006 IEEE.

[3, 9, 10], images, books [11], movies [12, 13], news, and homepages [7, 14, 15]. Depending on the system, the algorithm uses a pre-defined profile or user rating history to make its choices. Most user-based recommender systems focus on grouping users with similar interests [7-9], although some do try to match the preferences of single users according to their rating histories [1, 6].

Recommender systems play a role to use multiple mapping techniques to connect item and user layers, requiring accurate and appropriate pre-processing and presentation of items for comparison and matching. Item representations can consist of keyword-based profiles provided by content providers or formatted feature descriptions extracted by information retrieval techniques. Accordingly, item feature descriptions in recommender systems can be keyword- or content-based. Features for items, such as movies or books, are hard to extract because movies are composed of various kinds of media [6] and content analysis of books encounters the problem of natural language processing. Their keyword-based profiles are often determined by content providers. However, current image and audio processing techniques now allow for programmed extraction of content-based features represented by factors that include tempo and pitch distribution for music and chroma and luminance distribution for images.

Previous recommender systems can be classified in terms of content-based filtering versus collaborative filtering. Standard content-based filtering focuses on classifying and comparing item content without sharing recommendations with others identified as having the same preferences. Collaborative filtering method focuses on how users are clustered into several groups according to their preference. To avoid drawbacks associated with keyword-based searching (commonly used for online movie or book store databases), other designers emphasize content-based filtering focusing on such features as energy level, volume, tempo, rhythm, chords, average pitch differences, etc. Many music recommender system designers acknowledge drawbacks in standard collaborative filtering approaches—for instance, they can't recommend two similar items if one of them is unrated. To address the shortcomings of both approaches, some systems use content features for user classification and other systems find out group users with similar tastes [7, 16].

To address challenges tied to human emotion or mood and solve the sparsity problem of collaborative filtering method, some music and image retrieval system designers use IEC to evaluate item fitness according to user parameters [4, 5]. We adopted IEC for our proposed model, which uses agent evolutionary training for item recommendations. The results of our system tests indicate that trained agents are capable of choosing songs that match both user taste and emotion.

2.2. Interactive Evolutionary Computing

Genetic algorithm (GA) is an artificial intelligence system that allows for searches of solutions to optimization problems. According to GA construction rules, the structure of an individual's chromosome is designed according to the specific problem and genes are randomly generated once the system is initialized. Following GA procedures include 1) using a fitness function to evaluate the performance of various problem solutions, 2) selecting multiple individuals from current population, 3) modifying the selected individuals by mutation and crossover operators, and 4) deciding which individuals should be preserved or discarded for the next run; discarded solutions are replaced by new ones whose genes are preserved). A GA repeats this evolutionary procedure until an optimal solution emerges. The challenge of music recommendation was defining a fitness function that accurately represents subjective human judgment. Only then can such a system be used to make judgments in art, engineering, and education [4, 5].

Interactive Evolutionary Computing (IEC) which is an optimization method can meet the need of defining a fitness function by involving the human preferences. IEC is a GA technique whose fitness of chromosome is measured by a human user [18]. The main factor affecting IEC evaluation is human emotion and fatigue. Since users cannot make fair judgments when processing run evaluations, results will change for different occasions according to the user's emotional state at any particular moment. Furthermore, since users may fail to adequately process large populations due to fatigue, searching for goals with smaller population sizes within fewer generations is an important factor. Finally, the potential for fluctuating human evaluations can result in inconsistencies across different generations [19].

3. Using Evolutionary Agents for a Music Recommender System

3.1. Model Description

In our model, intermediate agents play the roles which select music compositions according to their chromosome and recommend to user. The system's six function blocks (track selector, feature extractor, recommendation agent module, evolution manager, user interface, and database) are shown in Figure 1.

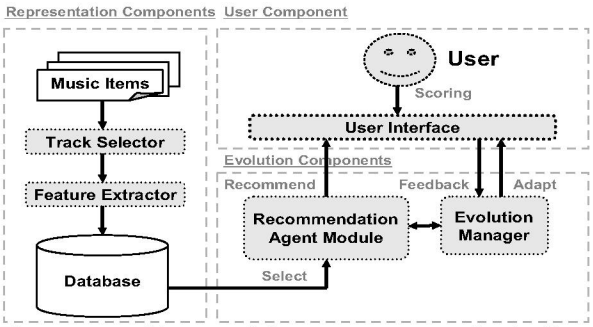

Figure 1. Six model components including track selector, feature extractor, database, recommendation agent module, evolution manager, and user interface

A representation component consists of the track selector, feature extractor, and database function blocks, all of which are responsible for forming item feature profiles. This component translates the conceptual properties of music items into useful information with specific values and stores it in a database for later use. In other words, this is a pre-processing component. Previous recommender systems established direct connections between user tastes and item features. In contrast, we use trainable agents to automatically make this connection based on a detailed item analysis. The track selector is responsible for translating each music composition into textual file, while feature extractor is responsible for calculating several statistical feature measurements (such as pitch entropy, pitch density, and mean pitch value for all tracks mentioned in Section 4). Finally, database function block stores these statistical features for further uses.

An evolution component includes a recommendation agent module and evolution manager. The former is responsible for building agent selection rules according to music features extracted by the representation component, while the latter constructs an evolution model based on IEC and applies a GA model to train the evolutionary agent. In our proposed model, user evaluations serve as the engine for agent adaptation (Fig. 2).

A central part of this component is the recommendation agent module, which consists of the agent design and the algorithm for selecting items. The first step for standard GAs is chromosome encoding—that is, designing an agent's chromosomal structure based on item feature representations. In our proposed model, each agent has one chromosome in which each gene respectively represents one of feature value. The gene value represents item feature preference and the number of item features represents chromosome length. Each feature needs two genes to express the mean and range value. Take 3 agents' chromosomes listed in Figure 3 for example, fl_mean and fl_range represent the 1st agent's preference of tempo feature. It means that 1st agent prefers the tempo between 30 and 40 beats per minute. The 1st agent will select the songs which have the tempo 35 ± 5 bests per minute and velocities 60 ± 10. The value of gene also can be "Don't care". We also perform the real number mutation for each mean and range value, and one-point crossover for selected pair of agents' chromosomes.

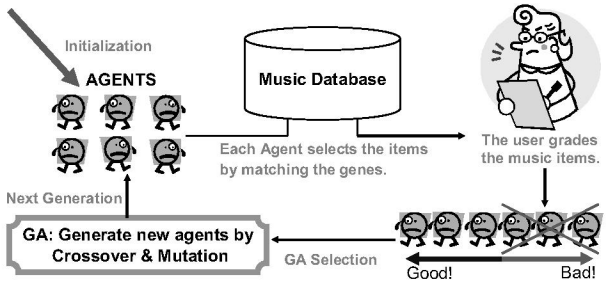

Figure 2. Evolution component, including agent recommendation module and evolution manager

The evolution manager in our model is responsible for the selection mechanism that preserves valuable genes for generating more effective offspring. The common procedure is selecting good agents to serve as the parent population, creating new individuals by mixing parental genes, and replacing eliminated agents. However, when dealing with subjective evaluations, human's preference changing can result in lack of stability across runs. Accordingly, the best agents in previous rounds may get low grades because of change of human's preference, and therefore be discarded prematurely. As a solution, we propose the idea of agent fame values that are established according to previous behaviors. The higher the value is, the greater the possibility that an agent will survive. The system's selection method determines which agents are discarded or recombined according to weighted fame values and local grades in each round, with total scores being summed with an agent's fame value in subsequent rounds.

Another important GA design issue is deciding when to stop agent evolution. System convergence is generally determined via learning curves, but in a subjective system this task (or deciding when an agent's training is complete) is especially difficult in light of potential change of user preference and emotion. Our solution is based on the observation that the stature of judges in a music or art competition increases or decreases according to decisions they

make in previous competitions. In our system, agent fame value varies in each round. The system monitors agent values to determine which ones exceed a pre-defined threshold; those agents are placed in a "V.I.P pool." Pool agents cannot be replaced, but they can share their genes with other agents. Once a sufficient number of stable V.I.P. agents are established, the system terminates the evolution process. For example, if one of agent got six points fame value and the system pre-define threshold is six points high, the agent will be placed in a V.I.P pool. This mechanism just sets for preserving the possible good agents.

CHROMOSOME

AgentID	f1_mean	f1_range	f2_mean	f2_range	...
1	35	5	60	10	...
2	60	3	95	4	...
3	83	5	120	10	...

Figure 3. Agent chromosome. Each gene represents a mean or range value of music feature. Whole chromosomes represent selection rules for agents to follow when choosing favorite items. The chromosome in this figure encodes two music features.

A user component consists of an interface for evaluating agent recommendations based on standards such as technicality, melody, style, and originality. The user interface is also responsible for arranging agents according to specific application purposes. For example, for finding joint preference between two different users, the user interface component will initialize and arrange two set agents for these two users respectively.

An agent selects items of interest from the database according to selection rules and makes appropriate recommendations to the user, who evaluates items via the interface. Evaluations are immediately dispatched to the agent, whose evolution is controlled according to performance and GA operations (e.g., crossover, mutation, and selection). The evolution manager is responsible for a convergence test whose results are used to halt evolution according to agent performance.

3.2. Applications

We designed our model so that the chromosomes of surviving agents contain selection rules that be able to represent user profiles. Concurrently, user profiles formed by agent chromosomes can be compared among multiple users. Combined, distributing agents can be utilized for three kinds of applications:

1. Users can train sample groups of agents. The agent evaluation function can be altered to reflect a sum of several user profiles, thus representing the tastes of multiple users. However, true system convergence will be difficult to achieve due to disagreements among user opinions. As in the case of scoring entries in art or music competitions, extremely high and low scores can result in total scoring bias.
2. Users can train their own agents and share profiles. According to this method (which is similar to collaborative filtering), the system compares user profiles formed by the agents' chromosomes and identifies those that are most similar. Collaborative recommendations can be implemented via partial exchanges among agents.
3. Users can train their own agents while verifying the items selected by other users' agents. In the art or music competition scenario, users can train their own agents before verifying the agents of other users to achieve partial agreement. Pools of agents from all users will therefore represent a consensus. If one user's choice is rejected by the majority of other users following verification, that user will be encouraged to perform some agent re-training or face the possibility that the agent in question will be eliminated from the pool. For this usage, the user interface is responsible for arranging and exchanging the agents between different users.

4. Experiments

Our experimental procedures can be divided into two phases:

1. 1. Training phase. Each user was allotted six agents for the purpose of selecting music items—two songs per agent per generation (12 songs per generation). Since subjective distinctions such as "good or bad music" are hard to distinguish according to a single grading standard, user give multiple scores to each songs according to difference standard. Each agent received two sets of scores from user, with three scores in each set representing melody, style, and originality. The chromosome of any agent receiving high grades from a user six times in a row was placed in the system's V.I.P pool; the chromosome was used to produce a new chromosome in the next generation. This procedure was repeated until the system determined that evolutionary convergence had occurred. The system stopped at the user's request or when the V.I.P pool contained four agents, whichever came first.
2. Validation phase. This phase consisted of a demonstration test for verifying that system-recommend songs matched the user's tastes. Experimental groups consisted of 20 songs chosen

by 6 trained agents; control groups consisted of 20 songs chosen by 6 random agents. User evaluations confirmed or refuted agent capabilities. Users were not told which selections belonged to the respective groups.

4.1. Model Implementations

Musical items were stored and played in polyphonic MIDI format in our system, because the node data in MIDI files can be extracted easily compared with data in audio wave format [1]. The track selector translates each MIDI file into a textual format respectively; we list the beginning part of textual feature file in Table 1 for example. Polyphonic items consist of one track for melody and additional tracks for accompanying instruments or vocals. The melody track (considered the representative track) contains the most semantics. Since the main melody track contains more distinct notes with different pitches than the other tracks, it was used for feature extraction based on pitch density analysis. According to previous research [3], this method is capable of achieving an 83 percent correctness rate. Track pitch density is defined as Pitch *density* = *NP* / *AP*, where *NP* is the number of distinct pitches on the track and *AP* is the number of all possible distinct pitches in the MIDI standard. After computing the pitch densities of all targeted music object tracks, the track with the highest density was identified as the representative polyphonic track.

Table 1. Part of textual MID feature file

Unit	Length	At	Time	Track	Channel	Note	Velocity
314	53	1162ms	197ms	T4	C4	d2	68
319	50	1181ms	185ms	T3	C3	d4	71
321	48	1188ms	178ms	T3	C3	b3	74
...	...	...	...	...	...	...	...

Purpose of feature extractor is to extract features from the perceptual properties of musical items and transform them into distinct data. We focused on seven features for our proposed system; new item features should be also added when possible.

1. Tempo, defined as the average note length value derived from MIDI files.
2. Volume, defined as the average value of note velocities derived from MIDI files.
3. Pitch entropy: $PitchEntropy = -\sum_{j=1}^{NP} P_j \log P_j, where Pj = \frac{N_j}{T}$, where N_j is the total number of notes with a corresponding pitch on the main track and T is the total number of main track notes.
4. Pitch density, as defined earlier in this section.
5. Mean pitch value for all tracks.
6. Pitch value standard deviation. Large standard deviations indicate a user preference for musical complexity.
7. Number of channels, reflecting a preference for solo performers, small ensembles, or large bands/orchestras.

Genes in standard GA systems are initialized randomly. However, in our proposed system the random agents will probably fail to find items that match their genetic information because the distribution of extracted features is unbalanced. We therefore suggest pre-analyzing feature value distribution and using the data to initialize agent chromosomes. By doing so, it is possible to avoid initial agent preferences that are so unusual that they cannot possibly locate preferred items. Furthermore, this procedure prevents noise and speeds up agent evolution. Here we will use tempo as an example of music feature pre-analysis. Since the average tempo for all songs in our database was approximately 80 beats per minute (Fig. 4), a random choice of tempo between 35 and 40 beats per minute resulted in eventual agent replacement or elimination and a longer convergence time before convergence for the entire system. For this reason, average values in our system were limited: 60 percent of all initial tempo ranges deviated between 1 and –1 and 80 percent between 2 and –2. This led to a speeding up of the agent evolution process.

4.2. Recommendation Quality

Recommendation quality is measured in terms of precision rate and weighted grade. Precision rate is defined as *Precision_rate* = N_S / N, where N_S is the number of successful samples and N the total number of music items. Weighted grades equals to summation of M_i divided by N, where M_i represents music item grades and N the total number of music items. Users were given six levels to choose from for evaluating chosen items.

Users were asked to evaluate experimental and control group selections. Experimental group agents evaluated songs recommended by agents that they had trained and control group agents evaluated songs at random. After users completed their tests, the system calculates precision rates and weighted grades. Finally, the songs recommended by the trained agents had an average precision rate of 84 percent and average weighted grade of 7.38, compared to 58.33 percent and 5.54 for songs recommended by the random agents.

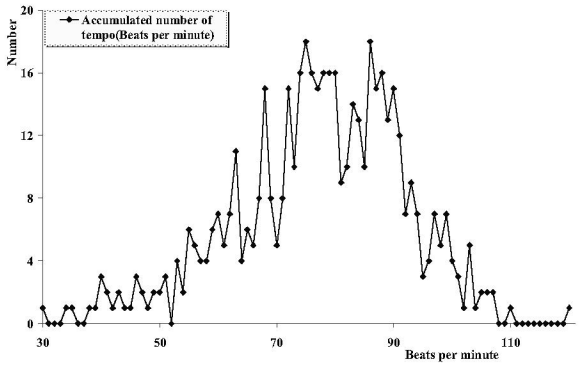

Figure 4. Statistical curve for tempo distribution in our sample of 1,036 MIDI files

4.3. Convergence Test

GA-based models commonly perform large numbers of iterations before arriving at convergence. In order to trace learning progress, we let users perform one demonstration test after every round; results are shown in Figure 5. Curve A reflects a steady increase in effectiveness and convergence after eight rounds. Curve B reflects a lack of progress for agents that make random selections without training.

In addition to recommendation quality and convergence tests, we made an attempt to identify clear differences between experimental and control group music selections by extracting their respective features. As shown in Figure 6, obvious differences were noted in terms of tempo and entropy, indicating that the trained agents converged unique preferences and did not blindly select items. Take one user's experimental result as an example, the user's preferences of feature tempo is quite different from the average tempo in control group.

5. Conclusion

Our proposed recommendation model can evaluate a large body of subjective data via a cooperative process involving both system agents and human users. Those users train groups of agents to find items that match their preferences, and then provide ongoing feedback on agent selections for purposes of further training. Agent training entails IEC methods and agent fame values to address the issue of change in human emotions. The agent fame value concept is also used as a convergence condition to promote agent population diversity and to propagate useful genes. Model flexibility was expressed in terms of replacing or altering functional blocks such as user interface which allows for usages of multiple users. We suggest that with refinement and modifications, our model has potential for use by referees to critique large numbers of subjective compositions (in such areas as art, music and engineering) and to make recommendations for images by extracting features (e.g., brightness, contrast, or RGB value) and encoding the information into agent chromosomes.

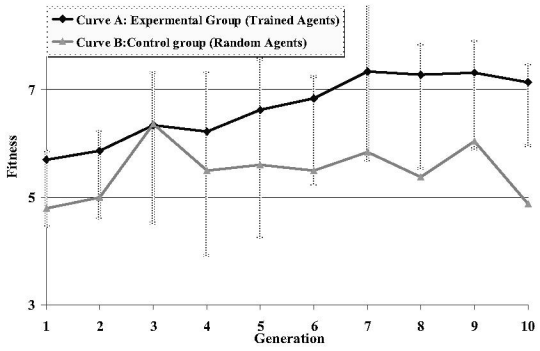

Figure 5. Convergence test and evolution generation of 10 users. Curve A represents an average of fitness values of 60 agents belong to 10 users

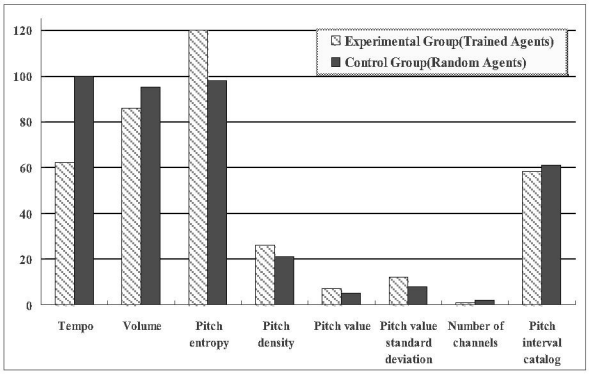

Figure 6. One user results example

References

[1] Kazuhiro, I., Yoshinori, H., Shogo, N.: Content-Based Filtering System for Music Data. 2004 Symposium on Applications and the Internet-Workshops. Tokyo Japan. (2004) 480

[2] Ben Schafer, J., Konstan, J.A., Riedl, J.: E-Commerce Recommendation Applications. Data Mining and Knowledge Discovery, Vol. 5. (2001) 115-153

[3] Chen, H.C., Chen, A.L.P.: A Music Recommendation System Based on Music and User Grouping. Journal of Intelligent Information Systems, Vol. 24. (2005) 113-132

[4] Cho, S.B.: Emotional Image and Musical Information Retrieval with Interactive Genetic Algorithm, Proceedings of the IEEE, Vol. 92. (2004) 702-711

[5] Cho, S.B., Lee, J.Y.: A Human-Oriented Image Retrieval System using Interactive Genetic Algorithm, IEEE Transactions on Systems, Man and Cybernetics, Part A, Vol. 32. (2002) 452-458

[6] Li, Q, Myaeng, S.H., Guan, D.H., Kim, B.M.: A Probabilistic Model for Music Recommendation Considering Audio Features, in Information Retrieval Technology, Vol. 3689. (2005) 72-83

[7] Balabanovic, M., Shoham, Y.: Fabs: Content-based, Collaborative Recommendation, Communication of the ACM, Vol. 40. (1997) 66-72

[8] Konstan, J.A., Miller, B.N., Maltz, D., Herlocker, J.L., Gordon, L.R., Riedl, J.: GroupLens: Applying Collaborative Filtering to Usenet News, Communications of the ACM, Vol. 40. (1997) 77-87

[9] Shardanand, U., Maes, P.: Social Information Filtering: Algorithms for Automating "Word of Mouth", In Katz, L.R., Mack, R., Marks, L., Rosson, M.B., Nielsen, J. (eds.), in Proceedings of the SIGCHI conference on Human factors in computing systems, Denver, Colorado, United States. (1995) 210-217

[10] Kuo, F.F., Shan, M.K.: A Personalized Music Filtering System Based on Melody Style Classification, in Proceedings of Second IEEE International Conference on Data Mining, (Maebashi City, Gumma Prefecture, Japan. (2002) 649-652

[11] Mooney, R.J., Roy, L.: Content-Based Book Recommending using Learning for Text Categorization, In Nurnberg, P.J., Hicks, D.L., Furuta, R. (eds.), in Proceedings of the fifth ACM conference on Digital libraries, (San Antonio, Texas, United States. (2000) 195-204

[12] Fisk, D.: An Application of Social Filtering to Movie Recommendation, Bt Technology Journal, Vol. 14. (1996) 124-132

[13] Mukherjee, R., Sajja, E., Sen, S.: A Movie Recommendation System - An Application of Voting Theory in User Modeling, User Modeling and User-Adapted Interaction, Vol. 13. (2003) 5-33

[14] Chaffee, J., Gauch, S.: Personal Ontologies for Web Navigation, in Proceedings of the ninth international conference on Information and knowledge management. McLean, Virginia, United States. (2000) 227-234

[15] Chiang, J.H., Chen, Y.C.: An Intelligent News Recommender Agent for Filtering and Categorizing Large Volumes of Text Corpus, International Journal of Intelligent Systems, Vol. 19. (2004) 201-216

[16] Pazzani, M.J.: A Framework for Collaborative, Content-Based and Demographic Filtering, Artificial Intelligence Review, Vol. 13. (1999) 393-408

[17] Holland, J.H.: Adaptation in Natural and Artificial Systems, Ann Arbor: University of Michigan Press. (1975)

[18] Takagi, H.: Interactive Evolutionary Computation: Fusion of the Capabilities of EC Optimization and Human Evaluation, in Proceedings of the IEEE, Vol. 89. (2001) 1275-1296

[19] Maes, P.: Agents that Reduce Work and Information Overload, Communications of the ACM, Vol. 37. (1994) 31-40