Evolving Color Paint

Mizutani, Eiji
Informations Systems Dept., Kansai Paint Co.,Ltd.

高木, 英行
Department of Acoustic Design, Kyushu Institute of Design

David M. Auslander
Department of Mechanical Engineering, University of California at Berkeley

https://hdl.handle.net/2324/4485873

出版情報：Proceedings of 1995 IEEE International Conference on Evolutionary Computation. 2, pp.533-538, 1995-11-29. IEEE
バージョン：
権利関係：

Evolving Color Paint

Eiji Mizutani [1], *Hideyuki Takagi* [2], *and David M. Auslander* [3]

eiji@joho.kansai.co.jp, takagi@kyushu-id.ac.jp, dma@euler.berkeley.edu

1) Informations Systems Dept., Kansai Paint Co.,Ltd, 4-3-6 Fushimi Chuo, Osaka 541, Japan
2) Acoustic Design Dept., Kyushu Institute of Design, 4-9-1 Shiobaru, Minami-ku Fukuoka 815, Japan
3) Dept.of Mechanical Engineering, University of California at Berkeley, CA 94720, USA

ABSTRACT

This paper presents an evolutionary computing intelligence for the paint industry. It integrates three principal components of soft-computing and a problem-specific knowledge to perform a computerized color recipe prediction; that is, neural networks (NNs), a fuzzy system (FS), and a genetic algorithm (GA) with a knowledge base (KB) complement each other in obtaining precise prediction outputs through the simulation of color paint manufacturing process.

In light of both potential versatility and practical validity in this industrial application, we have realized that the proposed hybrid system can yield advantages over other approaches in that it can evolve their results.

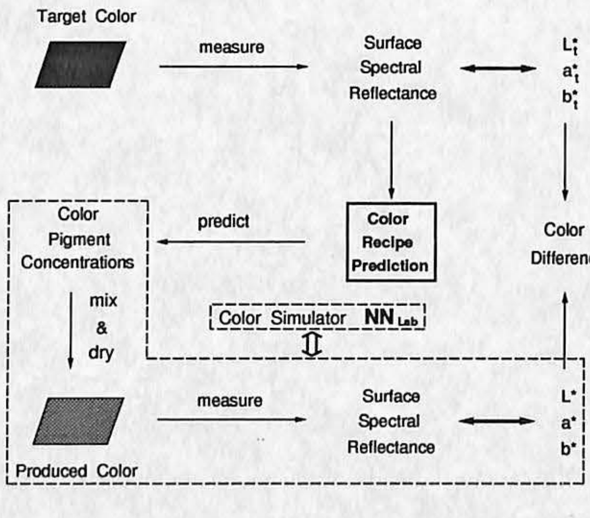

Fig. 1: Color paint manufacturing process.

1. Introduction

A color recipe prediction task is to predict which color pigments and what concentrations of the pigments are needed to produce the same color as a given reference color (see Figure 1). It is very difficult even for professional colorists to do well, yet human color perception is very sensitive and therefore the color matching must be done very well to meet acceptable standards. Since the widely-used Kubelka-Munk theory requires some assumptions which limit the situations where the theory may be applied [10], simple NN approaches, just like an NN in Figure 3 (A), have been introduced to overcome the practical obstacles in this problem area [2, 9]. The major concerns are (1) how to keep the practical number of necessary pigments, (2) how to consider the magnitude of the perceived color difference in the human visual sense, and (3) how to obtain precise pigment concentrations with enough precision to specify levels such as 0.01%. The first concern is a kind of combinational problem for which the GA may be a good choice. We use NNs to cope with the second critical problem. In regard to the third concern, we use a fusion technique of NNs, an FS, and a GA with a KB, to evolve color pigment concentration vectors. We shall clarify the evolutionary mechanism in the subsequent sections.

The authors would like to thank Mr. Kenichi Nishio (Sony Corp.), Dr. J.-S. Roger Jang (MathWorks, Inc.), and Mr. Tohru Hirayama (Kansai Paint Co.,Ltd) for their valuable advice. This work was mainly conducted when author[1] was a graduate student at the University of California at Berkeley.

2. Hybrid System Architecture

In the initial stage, starting points for a GA search are set by a fuzzy population generator and a multi-elite generator using results from NN approaches. In the evolutionary stage, the hybrid system tries to improve those results, which are already within some range of the ideal pigment concentrations. Two different NNs and a KB are used to make up the fitness function. Genes' pigment concentrations are passed to three functions which calculate fitness values individually. The three values are combined into the final fitness value. This mechanism is shown in Figure 2.

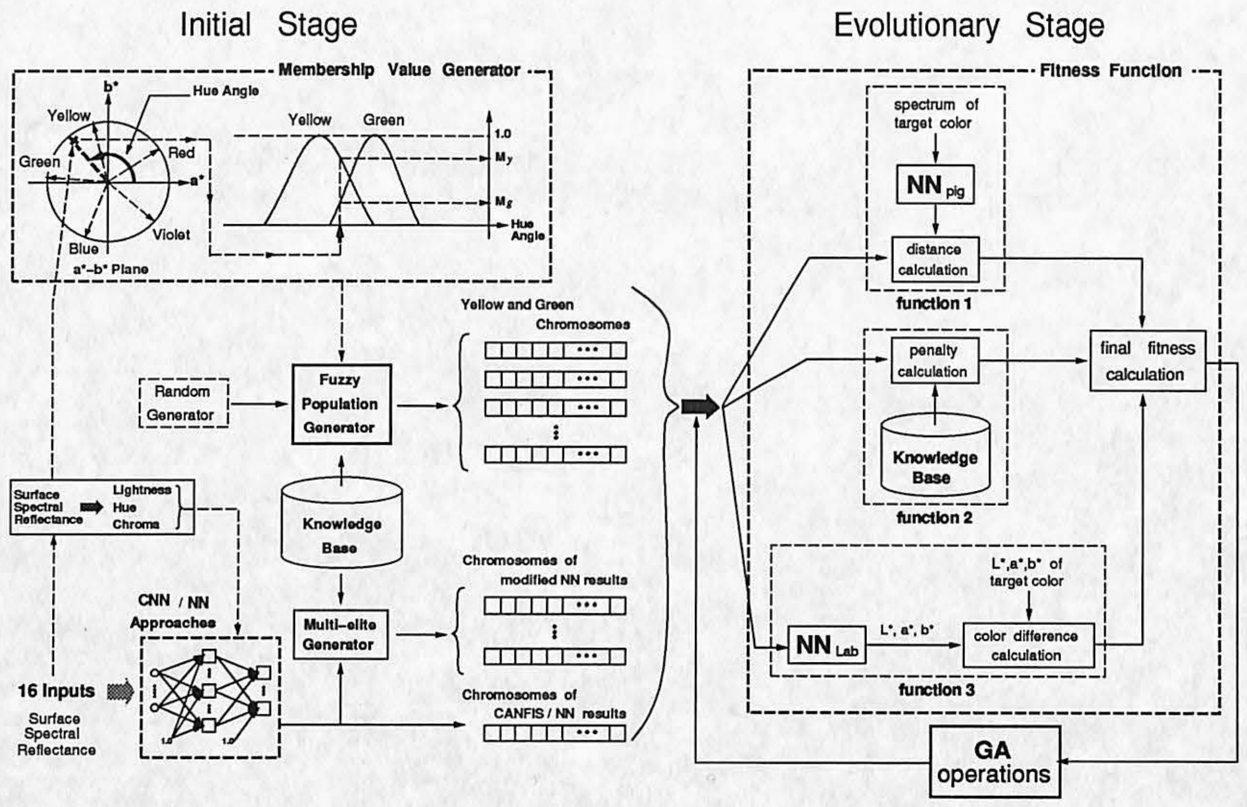

Fig. 2: Computational intelligence for color recipe prediction.

2.1. Knowledge Base

Knowledge may be useful in reinforcing some favorable aspects of genetic searches [6]. Performing the color recipe prediction task requires special knowledge; we believe that a KB plays an important role in helping the system evolve to recognize specific features of a target color. The KB has the following main rules:

Rule 1:	Keep total proportions of color pigments around 100 %,
Rule 2:	Keep the number of necessary color pigments around the ideal number,
Rule 3:	Avoid use of complementary color pigments: e.g., Red and Green,
Rule 4:	Avoid use of the same kind of color pigments at the same time.

We have 10 color pigments (10 outputs), including 3 pairs of the same kind of colorants: green, yellow, and red ones (see Figure 3). Note that Red_1 and Red_2, for example, have different characteristics. In this prediction task, the 100 % rule (Rule 1) was emphasized in [9], also.

2.2. Multi-elites Generator

Firstly, the results of NN approaches are encoded into initial population as elite members. Then, a multi-elite generator produces more elites by modifying those results according to Rule 4 in the KB. Namely, the concentrations of the same color pigments are summed into one or another of them: e.g., $\mathrm{Red}_1 + \mathrm{Red}_2 \Rightarrow \mathrm{Red}_1$, or $\mathrm{Red}_1 + \mathrm{Red}_2 \Rightarrow \mathrm{Red}_2$. This is derived from the fact that the simple NN tends to specify use of more than six pigments, though the desired number of pigments to produce any color in our data sets is fewer than five. Again, it is important to keep the number of pigments used at a practical level. Those multiple elite pigment vectors offer several different starting points for GA searches. The combination of several solutions may be effective in finding the optimal solution [5]. The other members are initialized by a fuzzy population generator. This seeding procedure is shown in Figure 2 (left).

2.3. Fuzzy Population Generator

The idea is to generate the initial population according to a fuzzy classification of a target color, which serves to determine color selection. First, we classify the target color into one of five color categories (red, yellow, green, blue, and violet) on the a*-b* plane, which shows hue and chroma [10] (see also Section 2.4.3), and decide to what extent the desired color belongs to each color category using fuzzy membership functions [7, 4]. We then generate initial color chromosomes by modifying randomly-generated chromosomes according to the rules in the KB. For example, when a target color looks greenish yellow, green chromosomes and yellow ones are generated; Green chromosomes have zero values in either $Green_1$ or $Green_2$ pigment concentration and in red pigment concentrations because of the red-green complementary color relationship. It is effective to inactivate some genes which have information on the same color and a complementary color in order to eliminate redundant pigments at the initial stage.

The number of green chromosomes (Num_{Green}) and that of yellow ones (Num_{Yellow}) are decided according to the following calculations:

$$\begin{aligned} Pop_{rest} &= Pop_{total} - Pop_{NN}, \\ Num_{Green} &= \frac{M_g \cdot Pop_{rest}}{M_y + M_g}, \\ Num_{Yellow} &= \frac{M_y \cdot Pop_{rest}}{M_y + M_g}, \end{aligned}$$

where two membership values, M_y and M_g, show to what extent the target color belongs to the yellow category and the green one, respectively. Pop_{total} denotes the total population number, and Pop_{NN} means the number of elite chromosomes from the NN results including the chromosomes generated by the multi-elite generator.

2.4. Fitness Function

The fitness function consists of three functions; two neural fitness functions, and the KB-based fitness function.

2.4.1. Function 1

Using NN_{pig}, the first function evaluates genes' pigment concentration vectors according to the specified use of pigments. The NN_{pig} ($16 \times 18 \times 21 \times 10$ neurons) maps surface spectral reflectance to a list of required pigments (see Figure 3(C)). It gives just ON/OFF values to each output unit to predict which pigments should be used to produce the same color as the target color, where ON means "needed" pigment and OFF means "not needed." Table 1 shows the capability of this trained NN_{pig}. Function 1 evaluates each chromosome by calculating a distance in binary space (ON/OFF) after each chromosome's representation has been transformed into the ON/OFF format.

2.4.2. Function 2

The second function calculates a fitness value based on the KB described in Section 2.1. The fitness value depends on the extent to which genes' pigment concentration vector obeys the rules in the KB. To keep the GA search moving in a consistent direction, the KB is used in both the initial stage and the calculation of fitness values as illustrated in Figure 2.

Table: 1: Capabilities of different NN approaches in specifying necessary pigments. CNN is a neuro-fuzzy model. NN_1 is a simple NN, and NN_2 is an improved NN_1; they appear in Figure 3(A). NN_{pig} is a special NN to predict necessary pigments as appeared in Figure 3(C).

	NN_1	NN_2	CNN	NN_{pig}
# of unmatched patterns in 302 test patterns	299	74	73	27
# of unmatched units in 3,020 output units	911	106	98	48
Predicted ave. # of required pigments to use	6.66	3.90	3.89	3.96

Table: 2: Average color difference predicted by NN_{Lab} using 302 checking data. The inputs of the NN_{Lab} were ideal pigment concentration vectors, and the results from three NN approaches; CNN is a neuro-fuzzy model, NN_1 is a simple NN, and NN_2 is an improved NN_1. This shows potential capability of the NN_{Lab}.

pigment vectors	ideal	CNN	NN_2	NN_1
color difference	0.567	1.976	2.847	5.921

2.4.3. Function 3

The third function, based on NN_{Lab}, generates a fitness value in terms of color difference between a target color and each member's color whose pigment concentrations are predicted by the system. Because it is time-consuming to manufacture an actual color by mixing genes' specified values (see Figure 1), the NN_{Lab} plays a crucial role as a color simulator to predict what color will be produced. The NN_{Lab} ($10 \times 11 \times 14 \times 3$ neurons) maps color pigment concentrations to L^*, a^*, and b^* (see Figure 3(B)). By plugging each member's pigment concentrations into NN_{Lab}, we can obtain predicted L^*, a^*, and b^*, to calculate the color difference between a target color and an individual color.

We adopted CIE 1976 (L^*, a^*, b^*)-space [10]. This defines the color difference and perceptual attributes of color, which are lightness, hue, and chroma as shown below:

Color Difference	$\sqrt{(L_t^* - L^*)^2 + (a_t^* - a^*)^2 + (b_t^* - b^*)^2}$
Lightness	L^*
Hue	$\arctan(b^*/a^*)$
Chroma	$\sqrt{(a^*)^2 + (b^*)^2}$,

where (L_t^*, a_t^*, b_t^*) are a target color values (see Figure 1). Note that L^*, a^*, and b^* are calculated according to surface spectral reflectance.

The calculated color difference shows how satisfactorily the predicted color matches the reference color. The use of NN_{Lab} provides a way to take into account human visual sensitivity to color difference. Table 2 shows the performance of the color simulator NN_{Lab}.

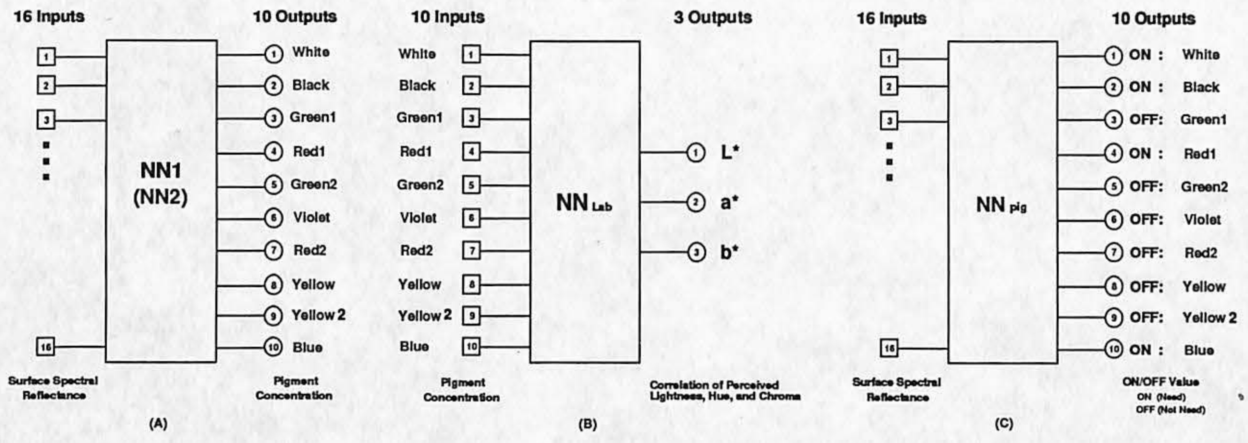

Fig. 3: Input/output relations of neural networks that are used in the proposed model.

3. Genetic Strategies

Genetic operations have a great impact on the quality of solutions. We have embodied some ideas in both mutation operations and crossover operations.

3.1. Mutation Strategy

Ordinary mutation operation as in a simple GA [3] is applied to all members with a fixed mutation rate scheme such that a fixed mutation rate (0.01) is adopted with a probability of 0.4, and otherwise, a mutation rate ranging from 0.09 to 0.69, is decided using a random number. Moreover, the following modified operations are also considered.

3.1.1. Chromosome Template

To avoid specifying use of more pigments than necessary, we set out to inactivate some genes using the fuzzy population generator as described in Section 2.3. This has made it possible to use a chromosome itself as a template to do the mutation operation. Namely, before the mutation operation, it is decided whether to mutate an inactivated gene or not; the mutation is applied with low probability (0.1) to inactivated genes, which have zero values of concentrations after decoding the genes' binary representations into pigment concentrations. If the mutation is applied to an inactivated gene, this leads to an increase in the number of necessary pigments.

3.1.2. Local Search and Preservation of Multi-elites

Multi-elites, i.e., chromosomes from the results of NN approaches and chromosomes originally generated by the multi-elite generator, are mutated only at the lower bits of each gene to keep traits similar to the NN results; Those mutant copies of the multi-elites can stay in the vicinity of the original multi-elites. In this way, local search of the NN results is realized. In addition, the offspring multi-elites always advance to the next generation; The mutant copies of multi-elites are preserved throughout the entire evolution. Note that this manipulation of low-order bits is applied only to the multi-elites.

3.1.3. Exchanging Mutation

After the usual mutation, members are subjected, with low probability, to another mutation: exchanging genes that have the same color information. This mutation is illustrated in Figure 4. Among the ten output pigment concentrations, we have three pairs of the same kinds of color pigments such as Red_1 and Red_2, which have actually different natures; we must decide which one to use. This exchanging mutation helps us to explore such color pigment choices. This can possibly lead to an escape from local optima in the initial NN and NN_{pig} results; their choices may not match the final choice determined by the system. The agreement with NN_{pig} in Table 4 shows how much the predicted choice optimized by the system matched the choices specified by NN_{pig}.

3.2. Modified Simplex Crossover

We have modified the selection method of simplex crossover [1]. Our modified strategy uses the following three procedures:

(1) select one good chromosome with respect to fitness value using fitness proportionate selection;
(2) select, with high probability, a multi-elite, as a good chromosome; and
(3) choose one bad chromosome with respect to fitness value.

This method may provide a better GA search direction as illustrated in Figure 5. When we have a

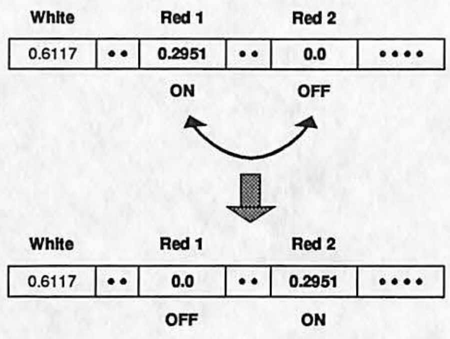

Fig. 4: Exchanging mutation

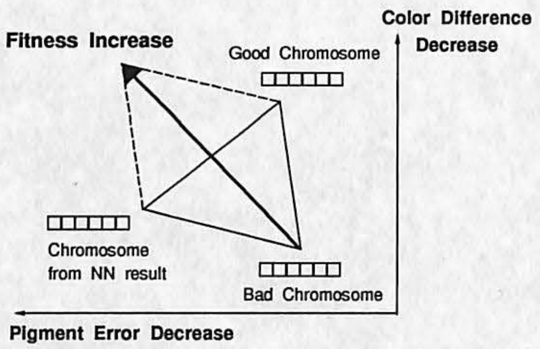

Fig. 5: GA search control by modified simplex crossover.

neural fitness function, we may have a problem; a trained NN may not be a perfect fitness function. Indeed, both NN_{pig} and NN_{Lab} in the fitness function are not perfect as shown in Table 1 and Table 2, but they may be able to direct a blind GA search to a better region of the search space. Procedure (1) lights a direction toward minimizing color difference. Thanks to NN_{Lab}, a chromosome with higher fitness may have smaller color difference. In this GA search, it is desirable to find a direction that minimizes both color difference and pigment errors. The problem is that we cannot calculate pigment errors directly. Yet the multi-elites provide a clue as to better pigment concentrations since they must already be within some range of the ideal pigment concentrations. That is why mutant copies from the multi-elites should be involved in guiding the paths towards better pigment concentration vectors as in procedure (2).

4. Experimental Results

The training data for NNs consist of 1,446 Munsell color (solid color) chips and the checking data of 111 standard paint color (solid color) chips of the Japan Paint Manufacturers Association. Both data sets are sampled from surface spectral reflectance at 16 points ranging from 400 nm to 700 nm in wavelength (20-nm intervals). The configuration of the GA was as follows:

Population size	80 members
Mutation rate	flexible (see Section 3.1)
Crossover method	simplex crossover [1]
Simplex Crossover Rate	0.85
Generation	10,000.

Table: 3: Results of computational prediction in pigment error ($\times 10^{-2}$) using 111 checking data.

NN_1	NN_2	CNN	GNF_{ALL}
2.312	1.543	1.139	0.643

Table: 4: Performance evaluation in computational prediction of 111 checking data. GNF_{ALL} shows the practical ability in the prediction task. A column, "agreement with NN_{pig}" implies how much the system match NN_{pig} in specifying pigments; and "color diff. by NN_{Lab}" shows the color difference predicted by NN_{Lab}.

	pigment error $\times 10^{-2}$	ave. # of pigments to use	agreement with NN_{pig}	color diff. by NN_{Lab}
GNF_{ALL}	0.643	3.90	79.28 %	0.267
(best)	0.213	3.88	79.28 %	0.809
GNF_{ALL}^{void}	72.209	3.94	50.45 %	48.800
(best)	36.165	4.89	21.62 %	34.637
GNF_{CP}	1.190	4.02	78.38 %	0.121
(best)	0.206	4.02	74.77 %	0.659
GNF_{CK}	1.695	3.88	74.77 %	0.202
(best)	0.215	3.89	77.48 %	0.656
GNF_C	2.802	5.35	28.83 %	0.060
(best)	0.191	4.36	55.86 %	0.567

Table 3 shows the comparison of our proposed hybrid system, GNF_{ALL}, and some other approaches; NN_1 was a simple NN approach, and NN_2 was an improved NN model that had modified sigmoidal functions in the output layer [8]. Both had the same model size ($16 \times 18 \times 21 \times 10$ neurons) (see Figure 3(A)). CNN was a neuro-fuzzy model [7]. GNF_{ALL} employed all three results from NN_1, NN_2, and CNN in producing initial population. To exhibit how indispensable those NN results were, we examined GNF_{ALL}^{void}, which had no multi-elites but had the same conditions as GNF_{ALL}.

Furthermore, to demonstrate the validity of each of three components in the fitness function, we tested GNF_C, GNF_{CP}, GNF_{CK} to compare our proposed model, GNF_{ALL}. GNF_C had NN_{Lab} as the only component of the fitness function. GNF_{CP} had both NN_{Lab} and NN_{pig} as two components of the fitness function. GNF_{CK} had a KB as well as NN_{Lab} as two components.function. GNF_{ALL} had all three components.

Note that NN_{Lab} has an important role as a color simulator, and so it always must stay in the fitness function. Table 4 shows how each component contributed to the prediction, and how they complemented each other. In Table 4, (best) means the best performance with respect to pigment errors to show a potential capability of each model, regardless of fitness; they were obtained when pigment errors were minimized.

5. Discussion

Both NN_{pig} and NN_{Lab} as components of the fitness function made some prediction errors as shown in Tables 1 and 2. Thus, enhancing the overall performance may result from improving the accuracy of such neural fitness functions.

Basically, the main focus in recipe prediction should be color difference rather than pigment errors. Therefore, the system should work in conjunction with NN_{Lab} to check predicted color difference. When NN_{Lab} alone as the fitness function was employed in GNF_C, the system tended to go too far toward minimizing color difference, and therefore the average number of required pigments was larger. Additionally, the specified pigments did not match very well those designated by NN_{pig} as indicated in the low percentage of agreement with NN_{pig} in Table 4. NN_{pig} was supposed to learn the characteristics of color pigment compositions such as complementary color relationships, yet it did not perform perfectly (see again Table 1). Thus, the KB surely helps the system evolve to recognize more features of color pigments. Here, it must be emphasized that they functioned complementarily.

GNF^{void}_{ALL} had no multi-elites but had all three components of the fitness function, starting from the randomly-initialized pigment concentration vectors. Its extraordinarily poor performance in predicted color difference places emphasis on the existence of the multi-elites, i.e., the mutant copies from other NN approaches; without them, we cannot draw any advantage from the search direction presented in Figure 5. In other words, seedings from other approaches are indispensable in enabling the system to function efficiently within a reasonable amount of computation time. Also, this poor performance may imply that fitness calculation should be altered for such no-seeding cases.

6. Conclusions

We have demonstrated both potential and practical strength of a unique blend of principal soft computing methods; a KB-based GA plays a leading role in pursuit of the prediction, linking an FS and NNs; they function complementarily as a system rather than competitively. The GA is not usually considered an ideal "fine-tuner" without specific modifications; here we have discussed a fusion technique as well as the modifications applied to the fitness function and GA operations. The GA alone may not be a good fine-tuner, yet the complete resulting system can be viewed as a fine-tuner, overcoming individual limitations. The entire hybrid system has realized higher precision of prediction by evolving the results from other approaches

The proposed system will be an incredibly promising tool to reach our goal, and it would possibly revolutionize an approach to another industrial application.

References

[1] Bersini, H. and Seront, G., "In search of a good evolution-optimization crossover," Parallel Problem Solving from Nature, 2 R.Manner and B.Mandrick (editors), Elsevier Science Publishers, 1992, pp.479-488.

[2] Bishop, J.M., Bushnell, M.J., Westland, S., "Application of Neural Networks to Computer Recipe Prediction," Color Research and Application, Vol.16, No.1, pp.3-9, (1991)

[3] Goldberg, D., *Genetic Algorithms in Search, Optimization, and Machine Learning*, Addison-Wesley (1989)

[4] Jang, J.-S. Roger, Sun, C.-T., and Mizutani, E., *Neuro-Fuzzy Modeling and Soft Computing,* Prentice Hall (in press)

[5] Kido, T., Kitano, H., Nakanishi, M., "A Hybrid Search for Genetic Algorithms: Combining Genetic Algorithms, TABU Search, and Simulated Annealing," 5th Int'l Conf. on Genetic Algorithms (ICGA'93), p.640 (July, 1993)

[6] Mansour, N., Fox, G.C., "A Hybrid Genetic Algorithm for Task Allocation in Multicomputers," 4th Int'l Conf. on Genetic Algorithms (ICGA'91), pp.466-473 (July, 1991)

[7] Mizutani, E., Jang, J-S. Roger., Nishio, K., Takagi, H., Auslander, D.M., "Coactive Neural Networks with Adjustable Fuzzy Membership Functions and Their Applications," Int'l Conf. on Fuzzy Logic, Neural Nets, and Soft Computing (IIZUKA '94), pp.581-582 (Aug., 1994)

[8] Mizutani, E., Takagi, H., Auslander, D.M., "A Cooperative system based on soft computing methods to realize higher precision of computer color recipe prediction," Applications and Science of Artificial Neural Networks, part of SPIE's Int'l Symposium on OE/Aerospace Sensing and Dual Use Photonics (April, 1995)

[9] Spehl, J., Wolker, M., Pelzl, J., "Application of Backpropagation Nets for Color Recipe Prediction as a Nonlinear Approximation Problem," IEEE Int'l Conf. on NN (ICNN'94), pp.3336-3341 (June, 1994)

[10] Wyszecki, G. and Stiles, W.S., *Color science: concepts and methods, quantitative data and formulae*, New York, NY: Wiley, 2nd ed. (1982)