Vol. 42 No. SIG 5(TOM 4) IPSJ Transactions on Mathematical Modeling and Its Applications May 2001

Regular Paper

Polynomial PAC Learnability of Learning Weights of Multi-objective Functions by Pairwise Comparison

KEN SATOH[†,☆]

This paper presents a theoretical analysis for a learning method of weights in linear combination of multi-objective function. Although there are several learning methods proposed in the literature[4),9),11),14),15),19)], none has yet been analyzed in terms of data complexity and computational complexity. This paper steps toward this direction of giving a theoretical analysis on learning method for multiple objective functions in the viewpoint of the computational learning theory. As the first step, this paper presents a theoretical analysis of learning method of weights from pairwise comparisons of solutions[14),15)]. In this setting, we show that we can efficiently learn a weight which has an error rate less than ϵ with a probability more than $1-\delta$ such that the size of pairs is polynomially bounded in the dimension, n for a solution, and ϵ^{-1} and δ^{-1}, and the running time is polynomially bounded in the size of pairs.

1. Introduction

In engineering domain, it is frequent that there are many objectives required to be optimal. For example, in making products, we have at least the following objectives:

(1) Shortening a duration of making products.
(2) Having lesser workers to make products.
(3) Decreasing burden of workers.
(4) Decreasing stocks of materials.
(5) Making products as soon as requests come.

However, it is rare that an optimal solution is obtained in which every objective takes an optimal value; we often encounter situations where some of the objectives conflict each other. In the above example, to shorten a duration of making products and to have lesser workers, we might have to increase burden of workers, and to make products just in time, we might have to store some stocks of materials beforehand.

In such situations, all we can hope is to obtain a Pareto solution in which a value of an objective cannot be enhanced without sacrificing a value of another objective. In other words, even if a Pareto solution x takes a maximal value $f(x)$ of some objective function f, x does not necessary take a maximal value $g(x)$ of some other objective function g and we cannot compare with other Pareto solution y such that $g(y)$ takes a maximal value and $f(y)$ does not.

Therefore, there are usually many Pareto solutions and therefore choosing a solution is not a easy task. To solve this problem, we can sometimes use user's *preferences* over objectives. In other words, if there are conflicting objectives, then we choose a preferable solution based on these preferences.

This situation can be regarded as an optimization problem for combination of objectives and preferences. But, this optimization problem is different from the ordinal optimization problem in operations research since in the above situation, an object function is unknown. One approach for this problem is to decide a single-valued objective function from multiple objective functions[4),9),11),14),15),19)]. Srinivasan and Shocker[14),15)] give a learning method of weights of linear combination of multi-objective functions from pairwise comparisons of solutions. The method has been applied to measurement of managerial success[16)] and preference of university administration[7)] and they show that this method is effective to some extent. Keeney and Raiffa[9)] give an analysis of multi-attribute utility functions under uncertainty and show conditions when the utility function has additive or multiplicative form and propose a method of deciding additive or multiplicative utility functions based on comparison of lottery. Dyer and Sarin[4)] modify the result by Keeny and Raiffa[9)] to give necessary and sufficient conditions of the form of multi-attribute value functions without uncertainty when the functions are additive or multiplicative and propose a method of deciding multi-attribute measurable value functions. Tamura

† Division of Electronics and Information Engineering, Hokkaido University
☆ Presently with the National Institute of Informatics

and Hikita[19] provide an interactive method based on decomposing a multi-attribute measurable value function into normalized conditional value functions and structural difference functions. Saaty[11] proposes a method called *analytic hierarchy process* (*AHP*) to decide a weight for each attribute by assessing consistency of preference value given by a user.

As far as we know, the above people only provide methods and evaluate empirically and there are no theoretical analyses in the viewpoint of data complexity and computational complexity. Although their contributions are important, it is also important to know whether their approaches are computationally feasible or not, since a system using the above methods must learn in a permissible amount of data and time.

This paper steps toward this direction of giving a theoretical analysis on learning method for multiple objective functions in the viewpoint of the computational learning theory. As the first step, this paper presents a theoretical analysis of learning method of weights from pairwise comparisons of solutions[14),15)]. In the work by Srinivasan and Shocker[14),15)], although they give a method which allows errors in comparison information and show empirical results, they do not give any theoretical results and therefore, it is not clear that how many pairwise comparisons are needed to achieve some accuracy of weights.

In this paper, imposing the condition that there are no error in comparison information and introducing probability distribution over solutions, we give a formal analysis of this method based on PAC (probably approximately correct) learning[20]. The contribution of this paper is to show that the learning method of weights in linear combination of objective functions using pairwise comparison is polynomially PAC-learnable.

The idea of the learning method of weights of objective functions from pairwise comparisons is as follows. Suppose that we have t objective functions $u_1(A), \ldots, u_t(A)$ for a solution A and the real function is a linear combination of these functions, that is, $\sum_{i=1}^{t} W_i * u_i(A)$. We compare two solutions A_1 and A_2 and whether it is more desirable or not. Suppose that A_1 is more desirable. This means that $\sum_{i=1}^{t} W_i * u_i(A_2) \leq \sum_{i=1}^{t} W_i * u_i(A_1)$. Since each W_i's must be consistent with these inequalities, possible range of W_i's will be limited as a set of inequalities grows. We can calculate such consistent W_i's by a linear programming algorithm.

Our analysis for the method is an extension of the analysis in learning weights in similarity function in case-based reasoning[12)] and learning preference relation in cardinality-based circumscription[13)]. In the former work[12)], we use relative distance information which tells if the distance between case A and case B is less than the distance between case A and case C and the algorithm learns weights in a weighted Euclidean distance of the cases. In the latter work[13)], we apply the above idea to learning preference relation for logical interpretations by regarding an logical interpretation as a case and a preference measure as a similarity measure between the interpretation and the most preferable interpretation. In this paper, we extend our previous results to learn weights of multi-objective functions of the more general form than those of both of previous works.

The paper is organized as follows. In Section 2, we informally explain the considered learning method and discuss the range of learnable preference functions. In Section 3, we give a formal definition of learning method and a theoretical analysis of the method in data complexity and computational complexity. In Section 4, we give a preliminary experimental result. In Section 5, we give conclusions and discuss future works.

2. Learning Preference Function by Pairwise Comparison

We explain the learning method by the following simple example. Suppose that every solution is represented as a point in R^2 and the preference function is represented as the distance from an ideal point I in some weighted Euclidean distance, that is, weighted additive measure for two objective functions where one objective function is the distance between the x component of I and that of the solution and the other is the distance between y component of I and that of the solution. The nearer solution to the ideal point is more preferable in this example.

In **Fig. 1**, we show two solutions and we assume that a user says that A is preferable to B. This information can be represented as $A < B$. If we use a usual (non-weighted) Euclidean distance function:

$$F(A, \langle 1, 1 \rangle) = (A_{(x)} - I_{(x)})^2 + (A_{(y)} - I_{(y)})^2,$$

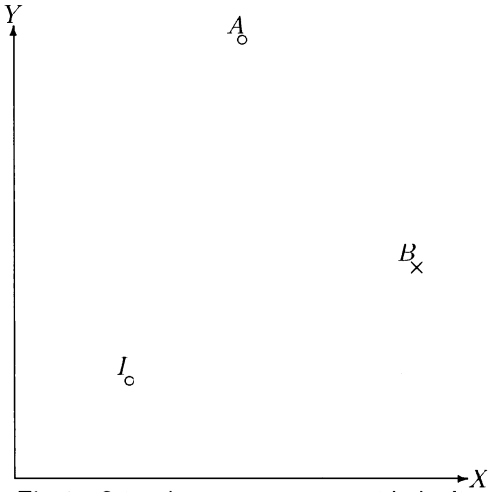

Fig. 1 Original space: inconsistent with the distance information $A < B$.

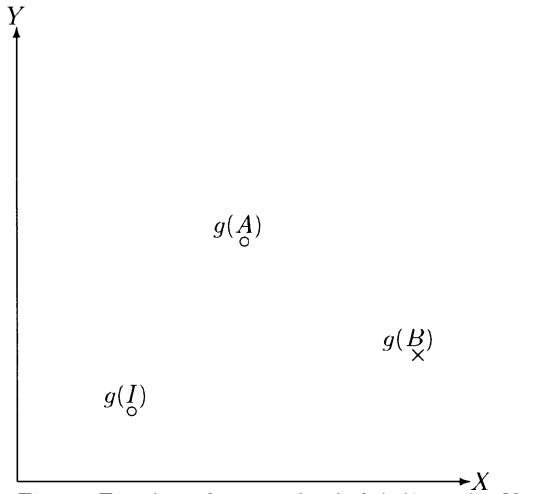

Fig. 2 Transformed space shrinked half in the Y-dimension: consistent with the distance information $A < B$.

where $\langle 1, 1\rangle$ is a weight vector for objective functions, then $F(A, \langle 1, 1\rangle) > F(B, \langle 1, 1\rangle)$ is true and the information from the user is contradictory. However, if we shrink Y dimension to a half, that is, we use the following distance function:

$$F\left(A, \left\langle 1, \frac{1}{4}\right\rangle\right) = (A_{(x)} - I_{(x)})^2 + \frac{1}{4}(A_{(y)} - I_{(y)})^2,$$

then the information from the user becomes consistent (**Fig. 2**). This distance function means that the importance of the objective function $(A_{(y)} - I_{(y)})^2$ is a quarter of that of the objective function $(A_{(x)} - I_{(x)})^2$. We would like to find such a proper transformation consistent with pairwise comparison. This corresponds with finding proper weights for objective functions.

In order to find proper weights, we set the following inequalities for every pairwise comparison.

For $A \leq B$,
$$((B_{(x)} - I_{(x)})^2 - (A_{(x)} - I_{(x)})^2) * W_{(x)} + ((B_{(y)} - I_{(y)})^2 - (A_{(y)} - I_{(y)})^2) * W_{(y)} \geq 0,$$
and for $B < A$,
$$((A_{(x)} - I_{(x)})^2 - (B_{(x)} - I_{(x)})^2) * W_{(x)} + ((A_{(y)} - I_{(y)})^2 - (B_{(y)} - I_{(y)})^2) * W_{(y)} > 0.$$

Note that a solution for $W_{(x)}$ and $W_{(y)}$ for the above inequalities is identical to a solution for $W_{(x)}$ and $W_{(y)}$ for the following inequalities.

For $A \leq B$,
$$((B_{(x)} - I_{(x)})^2 - (A_{(x)} - I_{(x)})^2) * W_{(x)} + ((B_{(y)} - I_{(y)})^2 - (A_{(y)} - I_{(y)})^2) * W_{(y)} \geq 0,$$
and for $B < A$,
$$((A_{(x)} - I_{(x)})^2 - (B_{(x)} - I_{(x)})^2) * W_{(x)} + ((A_{(y)} - I_{(y)})^2 - (B_{(y)} - I_{(y)})^2) * W_{(y)} \geq 1.$$

Therefore, by using linear programming, we can efficiently learn weights $W_{(x)}$ and $W_{(y)}$ which are consistent with the above information of the comparisons.

We extend learnable preference functions by the above method step by step. We can change the form of an objective function for each component as follows:

$$F(A, W) = \sum_{i=1}^{n} W_i * f_i(A_i).$$

Therefore, $f_i(A_i)$ can be a Euclidean distance from the ideal solution $(A_i - I_i)^2$ or a Manhattan distance $|A_i - I_i|$. This is the form of distance function considered in our previous work [12),13)].

We further extend the above objective function so that an objective function can be any arbitrary function $u_i(A)$ mapping a solution A to the real number.

$$F(A, W) = \sum_{i=1}^{t} W_i * u_i(A).$$

Note that it is possible that $t \neq n$.

We can also consider the arbitrary combination of objective function for a preference function such as:

$$F(A, W) = \sum_{i=1}^{t} W_i * u_i(A) + \sum_{1 \leq i \leq j \leq t} W_{ij} * u_i(A) * u_j(A).$$

As a general form, $F(A, W)$ can be:

$$F(A, W) = \sum_{i=1}^{m} W_i * F_i(u_1(A), \ldots, u_t(A)).$$

If we allow a preprocess of a preference function, we can make the following extension as well. For a preference function defined by the Minkowski metric:

$$F(A, W) = \left(\sum_{i=1}^{n} W_i |A_i - I_i|^r \right)^{\frac{1}{r}},$$

we consider the following function $F'(A, W)$ for making inequalities:

$$F'(A, W) = \sum_{i=1}^{n} W_i |A_i - I_i|^r.$$

Moreover, for a preference function of multiplicative form:

$F(A, W) = \Pi_{i=1}^{n} |A_i - I_i|^{W_i}$,

we take the logarithm of the above function and consider the following functions $F'(A, W)$ for making inequalities:

$$F'(A, W) = \log(F(A, W)) = \sum_{i=1}^{n} W_i * \log |A_i - I_i|.$$

3. Formal Analysis of the Learning Method

Let $A \in R^n$ be a solution and $u_i(A)$ $(1 \leq i \leq t)$ be objective functions such that t is bounded by a polynomial of n and $u_i(A)$ are calculated in the time bounded by a polynomial of n. Let preference function $F(A, W)$ be of the form

$$F(A, W) = \sum_{i=1}^{m} W_i * F_i(u_1(A), \ldots, u_t(A))$$

where m is bounded by a polynomial of n, and W is a weight vector $(W_1, \ldots, W_m)$, and F_i is a polynomially evaluatable function of $u_1(A), \ldots, u_t(A)$.

Note that since $u_1(A), \ldots, u_t(A)$ are calculated in the time bounded by a polynomial of n, each $F_i(u_1(A), \ldots, u_t(A))$ can be calculated in the time bounded by a polynomial of n. We assume that there exists a true weight vector $W^* = (W_1^*, \ldots, W_m^*)$.

Then, the learning problem is to find a hypothetical weight vector W which approximates W^* as possible. To do that, we provide our version of "approximation" of the true weight as follows.

We firstly define *a set of difference pairs between W and W^** as follows:

$$diff(W, W^*) \stackrel{def}{=} \{\langle A, B \rangle \in R^n \times R^n | (F(A, W) \geq F(B, W) \wedge F(A, W^*) < F(B, W^*)) \vee (F(A, W) < F(B, W) \wedge F(A, W^*) \geq F(B, W^*))\}$$

The above set consists of solution pairs (A, B) such that (1) a solution A is actually preferable to the other solution B, but from the hypothetical weight W, B is preferable to A or, (2) vice versa.

Let $\mathbf{P}$ be any probability distribution over n-dimensional Euclidean space, R^n. Then, W is said to be an *ϵ-approximation of W^* w.r.t. difference pairs for $\mathbf{P}^2$*, if the probability of $\mathbf{P}^2(diff(W, W^*))$ is at most ϵ where $\mathbf{P}^2$ is a distribution over $R^n \times R^n$ such that $\mathbf{P}^2(S \times T)$ is calculated from $\mathbf{P}(S) * \mathbf{P}(T)$ for sets $S \subseteq R^n$ and $T \subseteq R^n$ and $\mathbf{P}^2(diff(W, W^*))$ is calculated by $\Sigma_{i=0}^{\infty} \mathbf{P}^2(S_i \times T_i)$ where we decomose $diff(W, W^*)$ into a union of small sets $S_i \times T_i$ $(0 \leq i \leq \infty)$. We call ϵ an *error rate*.

The following theorem shows that this framework is polynomially PAC-learnable.

Theorem 1 There exists a learning algorithm which satisfies the following conditions for any probability distribution over R^n, $\mathbf{P}$, and an arbitrary constants ϵ and δ in the range $(0, 1)$:

(1) The teacher selects a true weight vector W^* from $[0, \infty)^m$.

(2) The teacher gives the definition of a preference function $F(A, W)$ with W unknown and gives N pairs according to $\mathbf{P}^2$ with the results of pairwise comparison defined by W^* to the algorithm.

(3) The algorithm outputs a hypothetical weight vector W and the following hold.
- The probability that W is not an ϵ-approximation of W^* w.r.t. difference pairs for $\mathbf{P}^2$ is less than δ. We call δ a *confidence*.
- The size of required pairs N for learning is bounded by a polynomial

in n, ϵ^{-1} and δ^{-1}, and so is its running time.

Proof: We firstly need the following definition.

Definition 1 Let w be a vector in $[0,\infty)^m$ and $\mathbf{P}'$ be a probability distribution over R^m. We say that w is an *ϵ-approximation of w^* w.r.t. difference points for $\mathbf{P}'$* if

$$\mathbf{P}'(\{x \in R^m | w \cdot x \leq 0 \text{ and } w^* \cdot x > 0\} \cup \{x \in R^m | w \cdot x > 0 \text{ and } w^* \cdot x \leq 0\}) \leq \epsilon$$

where $\cdot$ is the inner product of vectors.

According to the result by Blumer, et al. [1] for learning half-spaces separated by a hyperplane, there exists a learning algorithm which satisfies the following conditions for every distribution $\mathbf{P}'$ on R^m and every ϵ and δ in the range of $(0,1)$,

(1) The teacher selects w^* in $[0,\infty)^m$.

(2) The teacher gives a set X of N points according to $\mathbf{P}'$ with dichotomy (X^+, X^-) of X defined below:
for every $x \in X^+, w^* \cdot x \leq 0$, and
for every $x \in X^-, w^* \cdot x > 0$.

(3) The algorithm outputs a vector w such that the probability that w is not an ϵ-approximation of w^* w.r.t. difference points for $\mathbf{P}'$ is less than δ.

Since the VC dimension of this problem is m, according to Theorem 2.1 in Blumer et al's work [1], the number of required points N is at most

$$\max\left(\frac{4}{\epsilon}\log_2\frac{2}{\delta}, \frac{8m}{\epsilon}\log_2\frac{13}{\epsilon}\right) \quad (1)$$

Note that since m is bounded by a polynomial of n, N is bounded by a polynomial of n, ϵ^{-1} and δ^{-1}.

Any algorithm which produces consistent values of w with the following constraints:

$$\begin{array}{ll} \text{for every } x \in X^+, & w \cdot x \leq 0, \text{ and} \\ \text{for every } x \in X^-, & w \cdot x > 0. \end{array} \quad (2)$$

can be a learning algorithm.

We can use a linear programming algorithm (for example, Karmarkar's algorithm [8]) for the above algorithm by considering the following constraints:

$$\begin{array}{ll} \text{for every } x \in X^+, & w \cdot x \leq 0, \text{ and} \\ \text{for every } x \in X^-, & w \cdot x \geq 1. \end{array} \quad (3)$$

Since X^- is a finite set, if there exists a solution for the constraints (2), then there should be some positive number d such that for every $x \in X^-, w \cdot x \geq d$. Since the solution is not changed even if we multiply $\frac{1}{d}$ for both sides of $w \cdot x \geq d$, we can get the above constraints (3). Therefore, there exists a solution for the constraints (3) if and only if there exists a solution for the constraints (2) and the time of finding w is bounded by a polynomial of m, and therefore bounded by a polynomial of n.

Now, we consider the following function $g : R^n \times R^n \mapsto R^m$.

$$g(A,B) = (z_1, \ldots, z_m)$$
where $z_i = F_i(u_1(A), \ldots, u_t(A)) - F_i(u_1(B), \ldots, u_t(B))$ $(1 \leq i \leq m)$.

Let a set function $\mathbf{P}''$ over R^m be the following:

$$\mathbf{P}''(S') = \mathbf{P}^2(g^{-1}(S')).$$

Then, $\mathbf{P}''$ is a probability distribution over R^m.

Then, information of the pairwise comparison is equivalent to the following conditions:

$$\begin{array}{l} \text{for } A \leq B, W^* \cdot g(A,B) \leq 0 \text{ and} \\ \text{for } A > B, W^* \cdot g(A,B) > 0. \end{array} \quad (4)$$

Note that since each F_i is calculated in the time bounded by a polynomial of n, each z_i is also calculated in the time bounded by a polynomial of n. Therefore, the set of above inequality is constructed in the time bounded by the size of N and a polynomial of n, and therefore bounded by a polynomial of n, ϵ^{-1} and δ^{-1}.

From the above discussion, by using a linear programming algorithm, we can find W such that the probability that W is not an ϵ-approximation of W^* w.r.t. difference points for $\mathbf{P}''$ is less than δ with required points bounded by Eq. (1) and the time of finding W is bounded by a polynomial of n, ϵ^{-1} and δ^{-1}. Since if W is an ϵ-approximation of W^* w.r.t. difference points for $\mathbf{P}''$ then W is an ϵ-approximation of W^* w.r.t. difference pairs for $\mathbf{P}^2$, W is a wanted weight for the original problem. □

Figure 3 shows a learning algorithm of weights by binary comparison.

4. Experimental Results

We now show an experimental result under the following setting.

(1) $F(A,W)$ is defined as follows:

$$F(A,W) = \sum_{i=1}^{n} W_i * A_i.$$

In other words, $t = 1$, $u_1(A) = A$, $m = n$, and $F_i(u_1(A)) = A_i$.

(2) We use a randomized function to produce n values ranging over $(0,1)$ and regard it as a true weight vector W^*.

(3) We use a randomized function to produce

Learn(ϵ,δ,m)
ϵ: accuracy, δ: confidence, m: the number of weights in the preference function
begin
Receive the definition of the preference function $F(A,W)$ with W unknown, and $\max(\frac{4}{\epsilon}\log_2\frac{2}{\delta}, \frac{8m}{\epsilon}\log_2\frac{13}{\epsilon})$ pairs of solutions and the results of comparison from the teacher.
for every pair (A,B)
if $A \leq B$ **then** add the following inequality to the constraint set:
$F(A,W) \leq F(B,W)$
if $B < A$ **then** add the following inequality to the constraint set:
$F(B,W)+1 \leq F(A,W)$
Get consistent values for the above constraint set by linear programming and output W.
end

Fig. 3 Learning algorithm.

n values ranging over $(0,1)$ and regard it as a solution. We repeat this $2*N$ times to produce N pairs of solutions. We vary N from 1000 to 1000 in order to see influence of sample size over error rate.

(4) Using the above algorithm, we learn a weight vector by using a linear programming system, CPLEX [2].

(5) For the learned weight, we produce 10,000 test pairs randomly and calculate an error rate.

(6) We repeat 100 times above and take the average of error rates.

We use UltraSPARC-IIi (440 MHz) processor with 1 GB memory for experiments.

Figure 4 shows relationship between the number of objective functions and the square root of learning time of weights. Since the graph is almost linear, the order of the learning time is $O(n^2)$ where n is the number of objective functions. Readers should be aware that this result is not necessarily applicable for any distribution.

Figure 5 shows relationship between the number of objective functions and the error rate. The number of objective functions is almost proportional to the error rate.

Figure 6 shows relationship between the size of training pairs and the inverse of error rate. This graph is almost linear, so the size of training pairs is almost inversely proportional to the error rate.

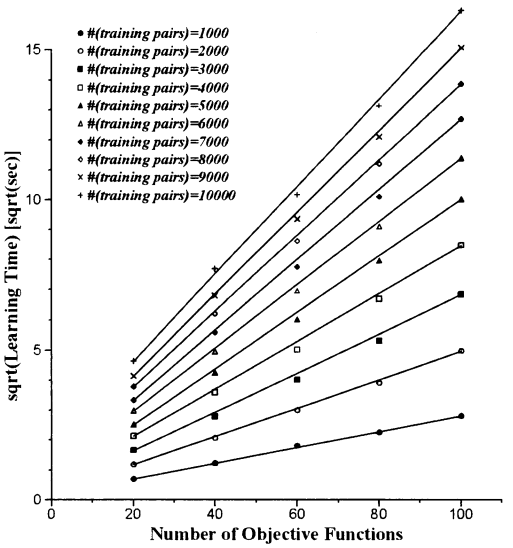

Fig. 4 Relationship between n and $\sqrt{LearningTime}$.

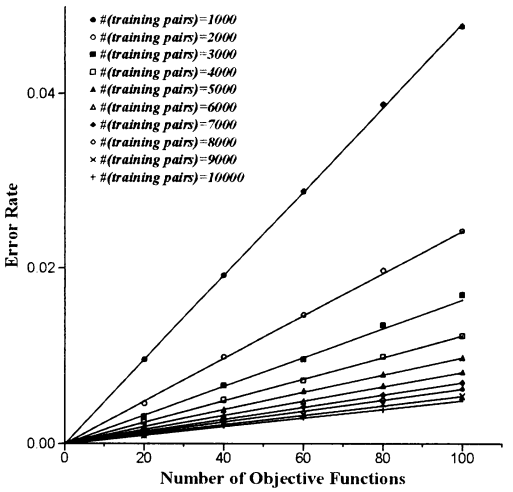

Fig. 5 Relationship between n and ϵ.

Figure 7 indicates that the required total size of training pairs is almost $\frac{C*n}{\epsilon}$ on average in order to obtain the average error rate, ϵ, where C is a constant. From the graph, $C \approx 0.488$. This size for training pairs is smaller than the size in our PAC-learning analysis. Actually, this is consistent with average-case analysis of the learning linear threshold function [5),6),18)]. For any fixed probability distribution of samples, Haussler, et al. [5),6)] show that the average error is bounded by $\frac{2n}{N}$, and

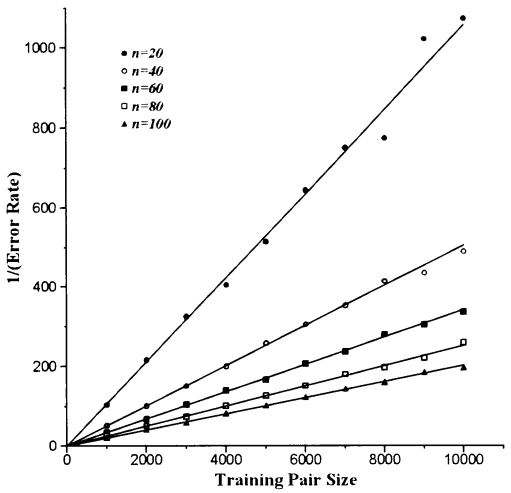

Fig. 6 Relationship between number of training pairs and ϵ^{-1}.

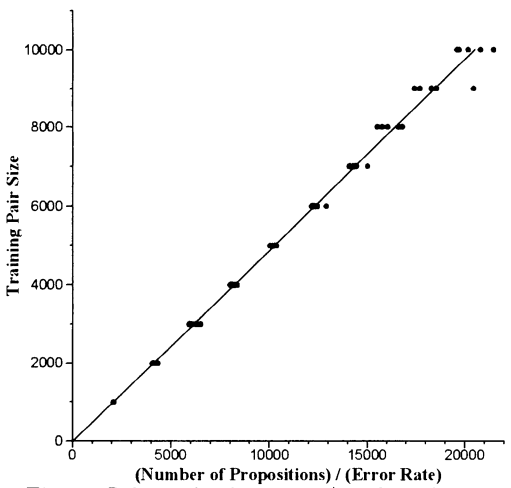

Fig. 7 Relationship between n/ϵ and number of training pairs.

Takahashi and Gu [18] shows that the average error is bounded by $\frac{n}{N+1}$ if $n \leq N$. Since our algorithm originates from a learning linear threshold function, the above result is very encouraging since the average sample pair size is bounded linearly to $\frac{n}{\epsilon}$ for any fixed probability distribution.

5. Conclusion

The contribution of this paper is a computational analysis of a learning method for weights in multi-objective functions which uses pairwise comparisons. The analysis shows that the learning method can polynomially PAC-learn the weight. Therefore, we can say that the learning method is feasible in terms of the worst-case analysis in computational learning theory.

We would like to pursue the following as future works.

- We would like to applying our method to real application domain to evaluate our analysis.
- Our analysis is based on the algorithm to get consistent hypothetical weights with comparison information. Therefore, if training data contains error or false comparison information, we cannot apply our analysis. Therefore, we need to extend our framework to consider errors on comparison to apply our analysis to real domain.

To solve the problem, we may use findings of Gu and Takahashi [17], and Cohen [3]. In the work of Gu and Takahashi [17], using their ill-disposed algorithm, they analyze the average error rate for any distribution of learning a linear threshold function with noisy data where the classification label of each example is flipped with some probability between 0 and $\frac{1}{2}$. In the work of Cohen [3], they show that learning a linear threshold function with a random classification noise is actually polynomially PAC learnable.

We would like to apply these methods for analyzing robust learning of weights.

- Speeding up of our algorithm is another direction of the research. In stead of using linear programming method, we may use the weighted-majority algorithm [10] or the support vector machine [21]. Analysis of learning weights using these methods is also an important future work.

Acknowledgments We are very grateful to Professor Hirotaka Nakayama from Konan University for the discussion of multi-objective optimization. We are also very grateful to Professor Haruhisa Takahashi from the University of Electro-Communications for teaching us average-case error bound and his algorithm on noisy data. We thank anonymous referees for instructive comments. This research is partly supported by Grant-in-Aid for Scientific Research (Project No. 11878067), The Ministry of Education, Japan.

References

1) Blumer, A., Ehrenfeucht, A., Haussler,

D. and Warmuth, M.K.: Learnability and the Vapnik-Chervonenkis Dimension, *JACM*, Vol.36, pp.929–965 (1989).
2) CPLEX Linear Optimizer 3.0, CPLEX Optimization, Inc. (1996).
3) Cohen, E.: Learning Noisy Perceptrons by a Perceptron in Polynomial Time, *Proc. 38th IEEE Annual Symposium on Foundations of Computer Science*, pp.514–523 (1997).
4) Dyer, J.S. and Sarin, R.K.: Measurable Multiattribute Value Functions, *Operations Research*, Vol.27, No.4, pp.810–822 (1979).
5) Haussler, D., Kearns, M., Opper, M. and Schapire, R.: Estimating Average-Case Learning Curves Using Baysian, Statistical Physics and VC dimension methods, *Advances in Neural Information Processing Systems*, Vol.4, pp.855–862.
6) Haussler, D., Kearns, M. and Schapire, R.: Bounds on the Sample Complexity of Bayesian Learning Using Information Theory and the VC Dimension, *Machine Learning*, Vol.14, pp.83–113 (1994).
7) Hopkins, D.S.P., Larrenche, J.C. and Massy, W.F.: Constrained Optimization of a University Administrator's Preference Function, *Manage. Sci.*, Vol.23, No.11, pp.1161–1168 (1977).
8) Karmarkar, N.: A New Polynomial-time Algorithm for Linear Programming, *Combinatorica*, Vol.4, pp.373–395 (1984).
9) Keeney, R.L. and Raiffa, H.: *Decision with Multiple Objectives, Prefernces and Value Tradeoffs*, John Wiley & Sons (1976).
10) Littestone, N. and Warmuth, M.K.: The Weighted Majority Algorithm, *Proc. 30th Annual Symposium on Foundations of Computer Science*, pp.256–261 (1989).
11) Saaty, T.L.: A Scaling Method for Priorities in Hierarchical Structures, *J. Mathematical Psychology*, Vol.15, pp.234–281 (1977).
12) Satoh, K. and Okamoto, S.: Learning Weights in a Similarity Function from Distance Information, *Journal of Japanese Society of Artificial Intelligence*, Vol.11, No.2, pp.238–245 (1996) (in Japanese). A preliminary English version appeared as Satoh, K. and Okamoto, S.: Toward PAC-Learning of Weights from Qualitative Distance Information, *Proc. 1994 AAAI Case-Based Reasoning Workshop*, pp.128–132, Seatle, USA (1994).
13) Satoh, K.: PAC-learning of Preference Relation over Interpretations in Lazy Nonmonotonic Reasoning, Motoda, H. and Muggleton, S. (Eds.), *Machine Intelligence 15*, pp.285–297, Oxford University Press (2000).
14) Srinivasan, V. and Shocker, A.: Linear Programming Techniques for Multidimensional Analysis of Preferences, *Psychometrika*, Vol.38, No.3, pp.337–369 (1973).
15) Srinivasan, V. and Shocker, A.: Estimating the Weights for Multiple Attributes in a Composite Criterion using Pairwise Judgments, *Psychometrika*, Vol.38, No.4, pp.473–493 (1973).
16) Srinivasan, V., Shocker, A. and Weinstein, A.G.: Measurement of a "Composite Criterion of Managerial Success", *Organizational Behavior and Human Performance*, Vol.9, pp.147–167 (1973).
17) Gu, H. and Takahashi, H.: Learning Curves in Learning with Noise – An Empirical Study, *IEICE Trans. Inf. & Syst.*, Vol.E80-D, No.1 pp.78–85 (1997).
18) Takahashi, H. and Gu, H.: A Tight Bound on Concept Learning, *IEEE Trans. Neural Networks*, Vol.9, No.6, pp.1191–1202 (1998).
19) Tamura, H. and Hikita, S.: An Interactive Algorithm for Identifying Multiattribute Measurable Value Functions based on Finite-Order Independence of Structural Difference (in Japanese), *Trans. SICE*, Vol.29, No.11, pp.758–764 (1985).
20) Valiant, L.G.: A Theory of the Learnable, *CACM*, Vol.27, pp.1134–1142 (1984).
21) Vapnik, V.: *Nature of Statistical Learning Theory*, Springer (1995).

(Received April 7, 2000)
(Revised July 6, 2000)
(Accepted September 5, 2000)

Ken Satoh was born in 1959. He graduated from the University of Tokyo in 1981 and joined Fujitsu Laboratories Ltd. He was sent to Institute for New Generation Computer Technology (ICOT) in 1987 and returned to Fujitsu Laboratories in 1992. He became an associate professor in Hokkaido University in 1995. Since 2001 he has been a professor in the National Institute of Informatics.