PAC Analysis of Learning Weights in Multi-Objective Function by Pairwise Comparison *

Ken Satoh
Division of Electronics and Information Engineering
Hokkaido University
North 13, West 8 Sapporo 060-8628, Japan
Email:ksatoh@db-ei.eng.hokudai.ac.jp
Fax: +81-11-706-7107 Tel: +81-11-706-7107

Abstract

This paper presents a theoretical analysis for a learning method of weights in multi-objective function. Although there are several learning methods proposed in the literature [Dyer79, Srinivasan73a, Srinivasan73b, Tamura85], 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[Srinivasan73a, Srinivasan73b].

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.

Keywords: machine learning, multi-objective function, PAC-learning, operations research, planning and scheduling

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.

*This is an extended paper of an abstract represented in *Proceedings of the Ninth International Conference on Industrial and Engineering Applications of Artificial Intelligence and Expert Systems*, p. 792, Fukuoka, Japan (1996).

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. However, there are many Pareto solutions and a user can sometimes say that some solutions are preferred to other solutions by his *preferences*. It is desirable if we can find more preferable solutions.

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 learn multiple objective functions [Dyer79, Srinivasan73a, Srinivasan73b, Tamura85]. It is very useful especially when we would like to transfer expert's preference into an expert system doing the above optimization problem in place of the expert. Unfortunately, however, the above researchers only provide methods and evaluate empirically, so there are no theoretical analyses in the viewpoint of data complexity and computational complexity. Although their contributions are excellent, it is very important 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[Srinivasan73a, Srinivasan73b]. The method has been applied to measurement of managerial success[Srinivasan73c] and preference of university administration [Hopkins77] and showed to be effective to some extent.

Our analysis for the method is an extension of the analysis in learning weights in similarity function in case-based reasoning[Satoh94] and learning preference relation in cardinality-based circumscription[Satoh95]. The analysis is based on PAC (probably approximately correct) learning [Valiant84]. In [Satoh94], 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 [Satoh95], 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 [Satoh94] and [Satoh95].

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

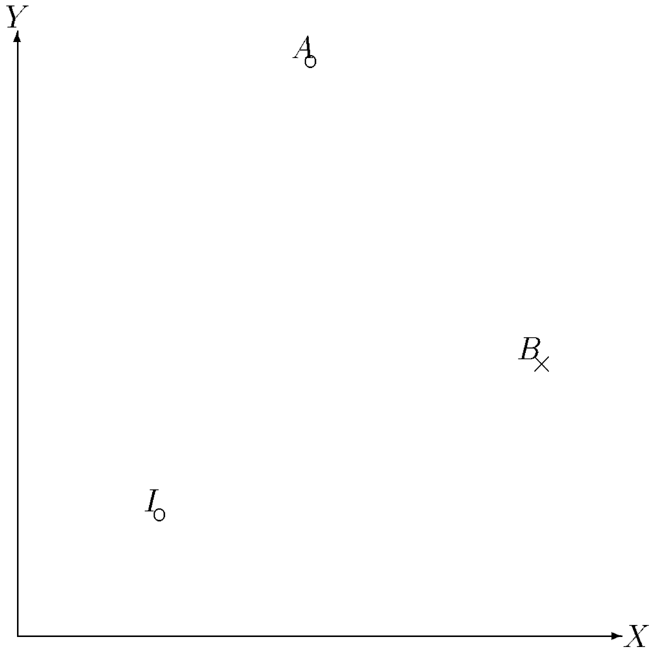

Figure 1: Original space: inconsistent with the distance information $A < B$.

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 Figure 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:

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

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:

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

then the information from the user becomes consistent (Figure 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

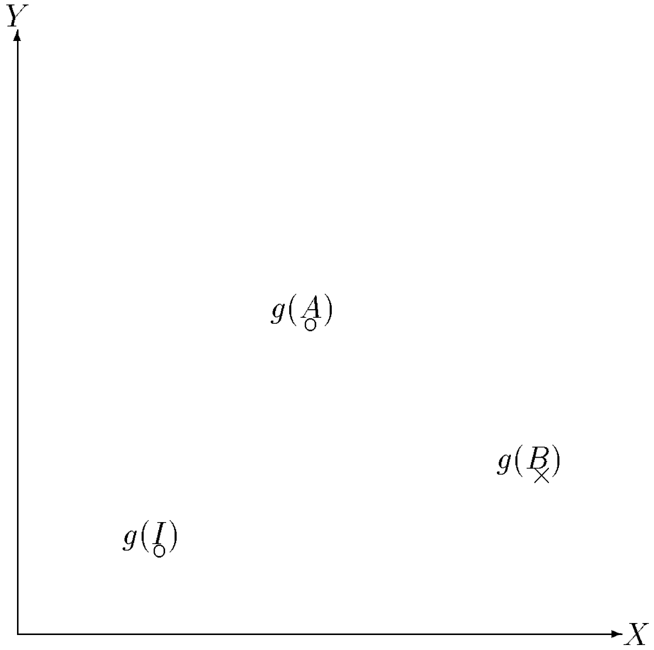

Figure 2: Transformed space shrinked half in the Y-dimension: consistent with the distance information $A < B$.

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 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 the learnable preference function 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 [Satoh94, Satoh95].

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 like:

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

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

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

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:

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

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

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

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

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

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.

Let $\mathbf{P}$ be any probability distribution over n-dimensional Euclidean space, R^n. Then, a set of different pairs between W and W^* is defined as follows:

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

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 hypothesis weight, B is preferable to A or, (2) vice versa.

W is said to be an *ϵ-approximation of W^* w.r.t. different pairs for* $\mathbf{P}^2$, if the probability of $\mathbf{P}^2(\mathit{diff}(W, W^*))$ is at most ϵ. 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. different 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: 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. different 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 in [Blumer89] 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:

$$\text{for every } x \in X^+, w^* \cdot x \leq 0, \text{ and for every } x \in X^-, w^* \cdot x > 0.$$

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

Since the VC dimension of this problem is m, according to Theorem 2.1 in [Blumer89], the number of required points N is at most

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

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:

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

can be a learning algorithm.

We can use a linear programming algorithm (for example, Karmarkar's algorithm[Karmarker84]) for the above algorithm by considering the following constraints:

$$\text{for every } x \in X^+, w \cdot x \leq 0, \text{ and for every } x \in X^-, w \cdot x \geq 1.$$

Clearly, there exists a solution for these constraints 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 $g : R^n \times R^n \mapsto R^m$,

$$g(A, B) = (z_1, ..., z_m)$$

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

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:

$$\text{for } A \leq B, W^* \cdot g(A, B) \leq 0 \text{ and for } A > B, W^* \cdot g(A, B) > 0. \quad (3)$$

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. different points for $\mathbf{P}$" is less than δ with required points bounded by (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. different 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.

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

Figure 3: Learning algorithm

4 Preliminary Experimental Result

We now show an experimental result under the following setting [1].

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 n values of 0 or 1 and regard it as a solution. That is, we consider the domain in which each value of the solution is boolean. We repeat this $2 * N$ times to produce N pairs of solutions.
4. For every pair, we produce the following relative preference information.

$$\begin{array}{l} \text{If } F(A, W^*) \leq F(B, W^*) \text{ then } A \leq B. \\ \text{If } F(A, W^*) > F(B, W^*) \text{ then } A > B. \end{array} \tag{4}$$

5. From (4), the algorithm learns a weight vector by using linear programming.
6. For the learned weight, we produce 10,000 test pairs randomly and calculate an error rate.
7. We repeat 100 times above and take the average of error rates.

[1]This is another interpretation of the experiment showed in [Satoh95].

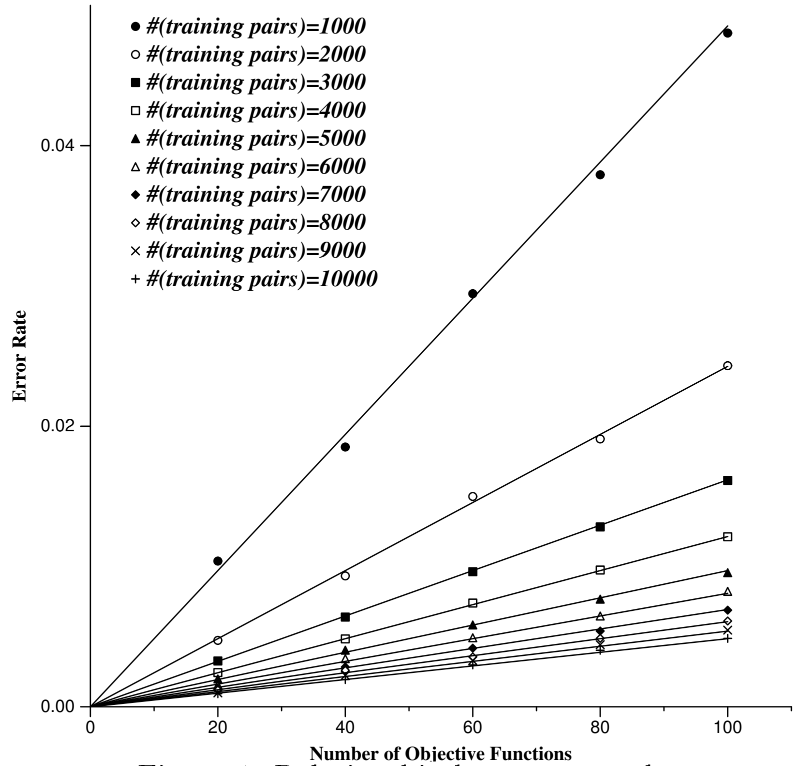

Figure 4: Relationship between n and ϵ

Figure 4 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 5 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.

Figure 6 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, $C \approx 0.485$. This size for training pairs is smaller than the size in our PAC-learning analysis. It is probably because in PAC-learning analysis, we consider the worst case, while in the experiment, we assume that the probability distribution is fixed where the behavior may not be so pathological.

5 Conclusion

This paper presents 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 pursue the following as future works.

1. Applying our method to real application domain.
2. Extending our framework to other kinds of metrics.

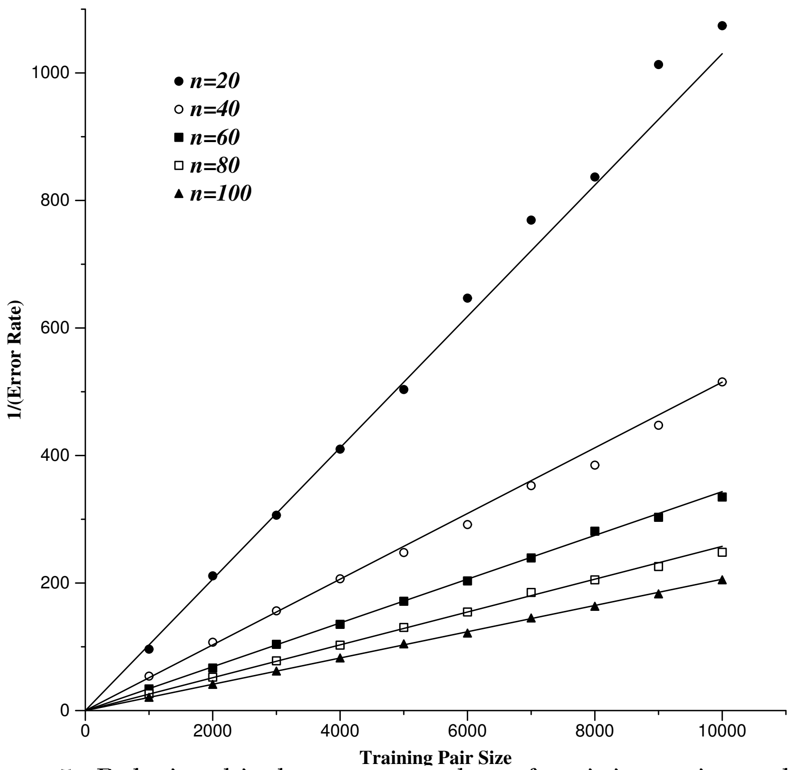

Figure 5: Relationship between number of training pairs and ϵ^{-1}

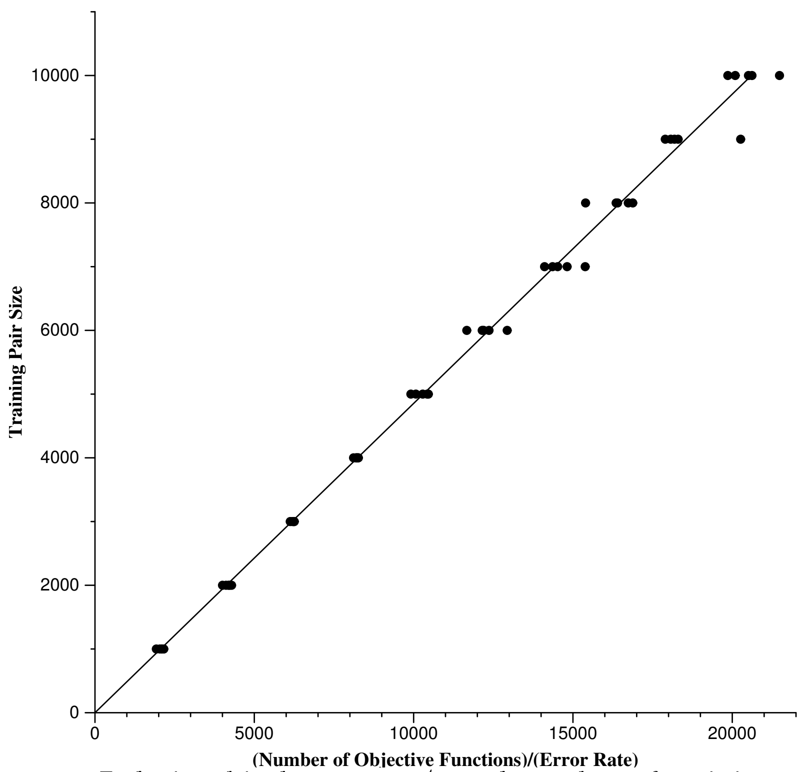

Figure 6: Relationship between n/ϵ and number of training pairs

3. Analyzing our method theoretically when the distribution is fixed to explain the experiment result in this paper.

Acknowledgements

I am very grateful to Professor Hirotaka Nakayama from Konan University for the discussion of multi-objective optimization and an anonymous referee for the improvement of the paper.

References

[Blumer89] Blumer, A., Ehrenfeucht, A., Haussler, D., and Warmuth, M. K., Learnability and the Vapnik-Chervonenkis Dimension, *JACM*, **36**, pp. 929 – 965 (1989).

[Dyer79] Dyer, J. S., and Sarin, R. K., Measurable Multiattribute Value Functions, *Operations Research*, Vol. 27, No.4, pp. 810 – 822 (1979).

[Hopkins77] Hopkins, D. S. P., Larrenche, J. C., and Massy, W. F., Constrained Optimization of a University Administrator's Preference Function, *Management Science*, Vol. 23, No.11, pp. 1161 – 1168 (1977).

[Karmarker84] Karmarkar, N., "A New Polynomial-time Algorithm for Linear Programming", *Combinatorica*, **4**, pp. 373 – 395 (1984).

[Satoh94] Satoh, K. and Okamoto, S., Toward PAC-Learning of Weights from Qualitative Distance Information, *Proc. of 1994 AAAI Case-Based Reasoning Workshop*, pp. 128 – 132 (1994).

[Satoh95] Satoh, K., PAC-learning of Preference Relation over Interpretations in Lazy Nonmonotonic Reasoning, to appear in *Machine Intelligence series*.

[Srinivasan73a] Srinivasan, V. and Shocker, A., Linear Programming Techniques for Multidimensional Analysis of Preferences, *Psychometrika*, Vol.38, No.3, pp. 337–369 (1973).

[Srinivasan73b] 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).

[Srinivasan73c] 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).

[Tamura85] Tamura, H. and Hikita, S., An Interactive Algorithm for Identifying Multiattribute Measurable Value Functions based on Finite-Order Independence of Structural Difference, Transactions of SICE (in Japanese), Vol 29, No. 11, pp. 62 – 68 (1985).

[Valiant84] Valiant, L. G., "A Theory of the Learnable", *CACM*, **27**, pp. 1134 – 1142 (1984).