PROCEEDINGS OF THE INTERNATIONAL CONFERENCE
ON FIFTH GENERATION COMPUTER SYSTEMS 1992,
edited by ICOT.

A Parallel Inductive Learning Algorithm for Adaptive Diagnosis

Yoichiro Nakakuki Yoshiyuki Koseki Midori Tanaka
C&C Systems Research Laboratories, NEC Corporation

4-1-1 Miyazaki, Miyamae-ku, Kawasaki 216, JAPAN
E-mail: nakakuki@btl.cl.nec.co.jp

Abstract

This paper describes a parallel inductive learning algorithm for adaptive model-based diagnosis. Although the model-based systems are more robust than the rule-based systems, they require more computation time. This is because they lack heuristic knowledge. On the other hand, human experts can learn and utilize such knowledge from experience. Therefore, in order to realize efficient model-based diagnosis, learning capability from experience is indispensable. We had proposed an inductive learning mechanism but unfortunately it took much computation time. In order to reduce the computation time, this paper proposes a parallel learning algorithm. The experiential knowledge is represented as a fault probability model and the proposed algorithm searches the most appropriate one out of all the possible models. In order to search effectively, a partial order is introduced into the search space. By using this ordering, two kinds of search control mechanisms, that are local pruning and global pruning, are developed. The algorithm is implemented in KL1 language on a parallel inference machine, Multi-PSI. The experimental results show the effectiveness of the mechanisms. It is also shown that the 16 PE implementation is about 11 times as fast as the sequential one.

1 Introduction

Since the creation of the MYCIN system[Shortliffe 1976], most of expert systems, have incorporated the idea of representing their knowledge in a form of symptom-failure association rules. Those expert systems that take rule-based approach have two major inherent disadvantages. First, those systems lack robustness because they cannot deal with unexpected cases which are not covered by rules in their knowledge bases. Second, their knowledge bases are expensive to be created and maintained.

There has been a series of research to tackle those problems. The most distinct ones are on model-based methods, i.e. first-principle methods. Model-based methods use design descriptions, such as structure and behavior descriptions [Davis 1984, de Kleer 1987, Genesereth 1984].

However, model-based diagnostic systems are generally not as efficient as rule-based ones since they require more complex computation. This is because they lack heuristic knowledge which human experts usually utilize. We have been working on a research to explore a general architecture to realize an adaptive diagnostic agent and introduced its basic architecture[Koseki 1989]. Moreover, an experimental system based on the architecture [Koseki et al. 1990a, Koseki et al. 1990b, Ohta et al. 1991a, Ohta et al. 1991b] have been developed. The system realizes adaptability with learning capability from its experience. The experiential knowledge is represented in a form of a fault probability model of target system components. With this experiential knowledge, it is able to diagnose a failing component faster with a fewer tests than pure model-based systems.

However, it takes much computation cost to learn experiential knowledge. This is because the hypothesis space to search grows rapidly with the size of the target problem. In order to reduce the computation time, we developed a parallel learning algorithm.

The algorithm utilizes two kinds of search control mechanism, that are local pruning and global pruning. The search space is divided and assigned to each processor so that the transmission of local pruning information does not require interprocess communication. The interprocess communication is restricted to the plausible global pruning information.

The algorithm is implemented in KL1 language on a parallel inference machine, Multi-PSI. The experimental results show that the implementation using 16 PEs is about 11 times as fast as the sequential one.

Section 2 presents the mechanism of the adaptive diagnostic system. In section 3, the probabilistic-model learning problem is described. A parallel learning algorithm is presented in section 4, and experimental results are shown in section 5.

2 Adaptive Diagnosis Mechanism

This section presents the architecture of an adaptive model-based diagnosis. We can observe two kinds of intelligent behavior in maintenance expert's diagnostic procedure. First, they can quickly identify a faulty component with a little information utilizing their experience. Second, even if a novel symptom arises, the expert can reach a conclusion, by consulting with other information sources, such as design description manuals. They can reason which component might have gone wrong and caused the symptom to appear, by knowing how the system is supposed to work.

To realize those kinds of intelligent behavior, the system consists of several modules as shown in Figure 2–1. The knowledge base consists of *design knowledge* and *experiential knowledge*. The design knowledge represents a correct model of the target device. It consists of structural description which expresses component interconnections and behavior description which expresses component behavior. The experiential knowledge is expressed as component failure probability for each component.

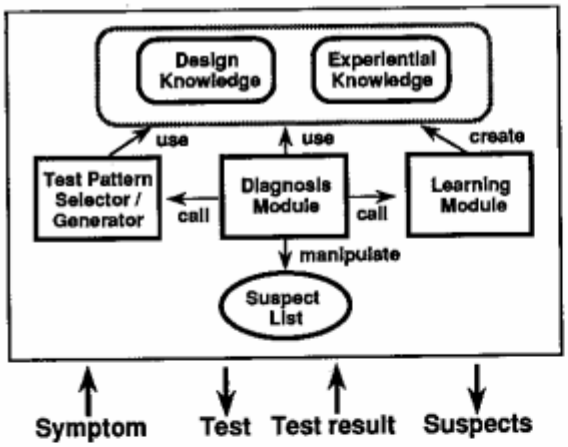

Fig. 2–1 Structure of the System

The general flow of the diagnostic system is shown in Figure 2–2. The system keeps a set of suspected components as a suspect-list. And it takes *eliminate-not-suspected* strategy[Tanaka et al. 1989] to reduce the number of the suspects in the suspect-list, repeating the test-and-eliminate cycle.

It starts with getting an initial symptom. It calculates an initial suspect list from the given initial symptom by performing a model-based reasoning. After obtaining the initial suspect-list, the system repeats a test-and-eliminate cycle, while the number of suspects is greater than one and an effective test exists. A set of tests is generated by the test pattern generator. Among the generated tests, the most cost effective one is selected as the next test to be performed. The selected test is suggested and fed into the target device. By feeding the test into the target device, another set of observation is obtained as a test result and is used to eliminate the non-failure components.

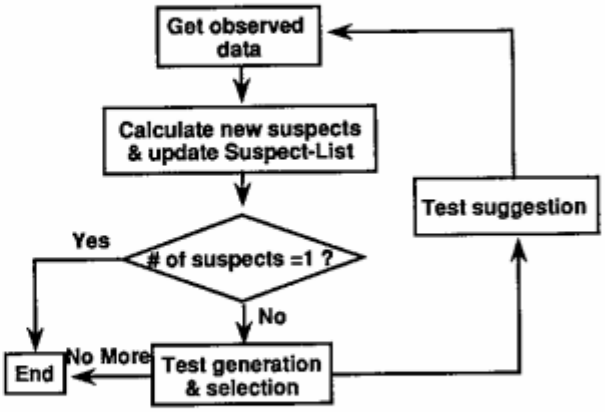

Fig. 2–2 Diagnosis Flow

In order to compute test effectiveness, the system uses fault probability distribution for each component. The mechanism employed in the system is basically same as the one found in the reference [de Kleer 1989]. It is so called *minimum entropy technique* where entropy is calculated from the fault probability for each suspected component. Here, an entropy $E(SL)$ of a suspect-list SL is defined in terms of the estimated probabilities of each component in the list. Let SL denote the set of suspected components,

$$SL = \{S_1, S_2, \ldots S_n\},$$

and let $p_1, p_2, \ldots p_n (\sum p_i = 1, \ p_i > 0)$ be failure probabilities of suspects $S_1, S_2, \ldots S_n$. Then an entropy $E(SL)$ is defined as

$$E(SL) = -\sum_{i=1}^{n} p_i \log p_i.$$

The system evaluates $gain(T)$ for all of the available tests. In addition to this value, the system considers the test execution cost to select a cost effective test. The system selects a test according to the following evaluation function.

$$gain(T)/cost(T)$$

At first, the diagnostic system does not know the probability distribution for a target device. Therefore, it should assume that the all of the components have the same fault probability. However, the system becomes efficient as it acquires information on the fault probability from its experience.This is because it can estimate more precise probability distribution and can generate more effective test sequence. In the next section, a learning mechanism is presented.

3 Learning Probabilistic Models

The performance of the diagnostic mechanism relies on the correctness of the presumed probability distribution of components. However, it is not easy to predict appropriate probability for each component from observed data, especially when the number of observed data is small.

For example, consider a diagnosis of a network system with 100 modems and 100 communication terminals. Here, we assume that 10 modems have broken down in the past (once for each). A simple estimate concludes that each of the 10 components has higher fault probability than any other component. However, human may presume that a modem has higher fault probability than a terminal because modems have broken 10 times in the past and terminals have never broken. Therefore, it is important to select an appropriate estimation method to derive a precise probability distribution (probabilistic model).

Here, we consider an example of a target device which consists of 16 components. The observed number of faults for each component is shown in Table 3–1. Several attributes for each component are also shown in the table.

Table 3–1 Example

Component	Attributes		No. of Obs. (Times)
	Type	Age	
1	a	new	1
2	a	old	0
3	b	new	13
4	b	old	9
5	c	new	1
6	c	old	1
7	d	new	0
8	d	old	0
9	e	new	0
10	e	old	0
11	f	new	1
12	f	old	0
13	g	new	0
14	g	old	5
15	h	new	1
16	h	old	0

First, we consider the relationship between the component type and the fault frequency. A type b component seems to have a very high fault probability. And it may be natural to conclude that type g component has also slightly higher probability than the other components. On the other hand, it is dangerous to conclude that each of the other components has different fault probability, e.g., the fault probability of type c component is about twice as large as type a component's. Because the difference between the number of observation may be due to an accident.

Next, we consider the relationship between component age and the fault probability. In the example, it seems that the component age does not affect the fault probability. Therefore, in order to estimate the fault probability distribution precisely, it is important to consider component type.

In general, some attributes are important to estimate the fault probability and the other attributes are not so important. Moreover, a combination of several attributes may be important. For instance, in the above example, we had better to consider component age, in the case of the component type is g.

In order to estimate the probability distribution precisely, we must find relevant attributes (and/or their combination) and consider how to estimate with those attributes.

Here we define the presumption problem. Consider a set of events $X = \{x_1, x_2, ..., x_m\}$ and attributes $a_1, a_2, ..., a_n$. Here, we assume that the events are exhaustive and mutually exclusive, and that the domain for each attribute a_j $(j = 1, 2, ..., n)$ is a finite set $Dom(a_j)$. As shown in Table 3–2, for each event, x_i, a value, v_{ij} $(\in Dom(a_j))$, for each attribute, a_j, is given. Also, n_i, the number of observations is given.

Table 3–2 Table of events

Event	Attributes				No. of Obs. (times)
	a_1	a_2	$\cdots$	a_n	
x_1	v_{11}	v_{12}	$\cdots$	v_{1n}	n_1
x_2	v_{21}	v_{22}	$\cdots$	v_{2n}	n_2
x_3	v_{31}	v_{32}	$\cdots$	v_{3n}	n_3
$\vdots$	$\vdots$	$\vdots$	$\ddots$	$\vdots$	$\vdots$
x_m	v_{m1}	v_{m2}	$\cdots$	v_{mn}	n_m

The problem is to presume the probability $\hat{p}_i$ for each event x_i, from the number of observations n_i. If enough amount of data are given, it seems to be easy to estimate the probability appropriately. However, if only a few observation data are given, we must consider the noise affection. Therefore, it is important to extract reliable information by avoiding the noise affection. In order to estimate the fault probability appropriately, we introduced an inductive learning mechanism [Nakakuki et al. 1990, Nakakuki et al. 1991b, Nakakuki et al. 1991c].

In the learning mechanism, a *presumption tree* is used to express a probabilistic model. Using a presumption tree, all the events are classified into several *groups*. Here, each event in a group is assumed to have the same fault probability. Therefore, the probabilities for individual events can be calculate from a presumption tree. The details are described below.

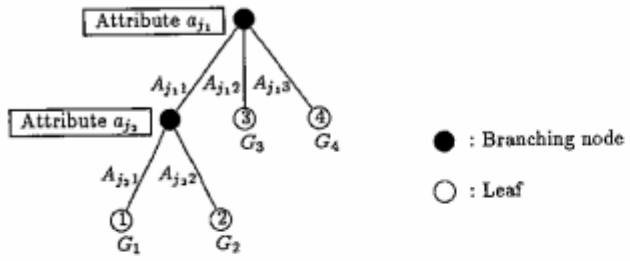

Fig. 3–1 Presumption tree

As shown in Fig. 3–1, a presumption tree consists of several branching nodes and leaves. An attribute a_j corresponds to each branching node, and subset A_{jk} of $Dom(a_j)$ corresponds to each branch. Here, each A_{jk} must satisfy the following conditions.

$$A_{ij} \cap A_{ik} = \phi \quad (j \neq k)$$

$$\bigcup_j A_{ij} = Dom(a_i)$$

A presumption tree classifies all possible events into several groups. For example, the tree in Fig. 3–1 has four leaves, therefore, the events are classified into four groups by using the tree as a decision tree[Quinlan 1986].

For example, G_1 is a group of events which corresponds to leaf 1. Each event in G_1 is considered to have the same fault probability. Here, for each leaf k, let its corresponding group of event be X_k, and for all event $x_i \in X_k$, let the sum of n_i be O_k. By using a presumption tree, the probability $\hat{p}_i$ for each event $x_i \in X_k$ can be calculated as follows:

$$\hat{p}_i = \frac{1}{|X_k|} \cdot \frac{\sum_{x_i \in X_k} n_i}{\sum_{x_i \in X} n_i}$$

As shown above, a presumption tree represents a probabilistic model. The problem is to find the most appropriate presumption tree for given data.

As a criterion for the selection, we introduced the minimum description length (MDL) criterion[Rissanen 1978, Rissanen 1983, Rissanen 1986]. Rissanen argued that the least description length model is expected to fit for presuming the future events better than any other models. Here, description length for a model is defined as the sum of the model-complexity and model-fitness for the given data. The description length of a presumption tree is the sum of:

(1) Code length of a tree, and

(2) Log-likelihood(distance) between the tree and observed data.

The code length(model complexity), L1, for a presumption tree is defined as follows[Nakakuki et al. 1991b].

$$L1 = \sum_{x \in P \cup Q} \log (n - d_x + 1) + \sum_{x \in Q} \frac{1}{2} \log O_x + \sum_{x \in P} \{\log (k_x - 1) + \log c1(k_x, l_x)\}$$

Here, P is a set of all the branching nodes and Q is a set of all the leaves. For each branching node x, l_x is the number of branches, d_x is the depth of the node, $k_x = |Dom(a_i)|$ (a_i is a corresponding attribute for node x), n is the number of attributes, and $c1(k_x, l_x) = \binom{k_x}{l_x - 1}$ if $l_x < k_x$, otherwise 1. On the other hand, log-likelihood (model fitness), L2, between a model and observed data is defined as follows.

$$L2 = -\sum_i n_i \log \hat{p}_i$$

Here, $\hat{p}_i$ is the presumed probability that is derived by using the model. The total code length is the sum of L1 and L2.

4 A Parallel Learning Algorithm

4.1 Local Pruning Mechanism

As described in the previous section, the problem is to search the least description length tree out of all the possible presumption trees. A heuristic algorithm for the problem was implemented[Nakakuki et al. 1991b] for a sequential machine by using branch-and-bound technique. The following summarizes the algorithm and then proposes a parallel version of the algorithm.

Here, let the length of a presumption tree T be denoted by $L(T)$. It is the sum of model complexity($L1(T)$) and the model fitness ($L2(T)$). Intuitively, a large tree has large model-complexity, and a small tree has large(bad) model-fitness[Nakakuki et al. 1991d]. In order to discuss such characteristics more precisely, we introduce a partial order "$\succ$" among the possible presumption trees. The order is defined as:

$T_2 \succ T_1 \underset{\text{def}}{\iff}$ Presumption tree T_2 can be obtained by replacing some leaves in presumption tree T_1 with branching nodes.

For example, presumption tree T_b in Fig. 4–1 can be obtained by replacing leaf z in T_a, therefore,

$$T_b \succ T_a.$$

Similarly,

$$T_c \succ T_a \text{ and } T_c \succ T_b.$$

Intuitively, $T_2 \succ T_1$ means T_2 is strictly larger than T_1.

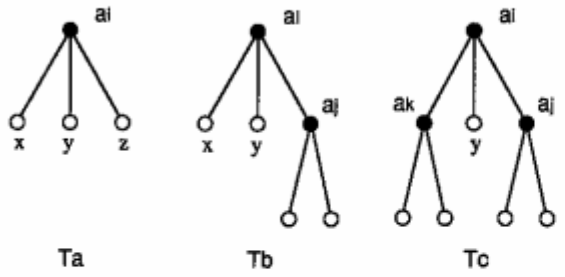

Fig. 4–1 Example

If $T_2 \succ T_1$, then the following inequalities hold by the definition:

$$L1(T_1) \leq L1(T_2)$$
$$L2(T_1) \geq L2(T_2)$$

Therefore, for a certain presumption problem, if a presumption tree T is a maximal one under the ordering, then $L2(T)$ will take the least value, say $L2_{MIN}$. $L2_{MIN}$ can be easily calculated in advance. By using these characteristics, we can effectively find a least description length tree.

The proposed algorithm searches the space of possible presumption trees. It tests simpler tree before testing more complex ones. That is, if there are two presumption trees T and T' such that $T' \succ T$, the system calculates the length of T before trying T'.

Here, consider that the length of a tree T has been tested. Then, the system considers the necessity of testing T' which is more complex than T (i.e. $T' \succ T$). If it turned out to be unnecessary(i.e., there is no possibility that T' has shorter length than T), then all the trees which are more complex than T' also turns out to be unnecessary to examine. The details of this technique are as follows.

In order to decide the necessity, the algorithm tests the following pruning condition:

$$\log(n - d_x + 1) + \log(k_x - 1) + \log\ c1(k_x, l_x)$$
$$+L2_{MIN} - L2(T) > 0$$

Here, x is one of the leaves in T and its corresponding node in T' is a branching node. If the inequality holds, it is not necessary to calculate the length for T'.

proof First, it is clear that the following inequality holds by the definition of L1:

$$L1(T') - L1(T)$$
$$\geq \log(n - d_x + 1) + \log(k_x - 1) + \log\ c1(k_x, l_x).$$

Second, the following inequality holds obviously:

$$L2(T') - L2(T) \geq L2_{MIN} - L2(T).$$

Here, if the sum of the right hand sides of the above two inequalities is positive(i.e., the pruning condition holds), then the sum of the left hand sides will be positive. Hence,

$$L1(T') + L2(T') > L1(T) + L2(T).$$
$$\text{i.e. } L(T') > L(T).$$

Therefore, it is not necessary to test T'. □

Here we consider to implement a parallel version of the algorithm. It is natural to divide the search space and to assign each sub-space to individual processor. However, we must be careful when we divide the search space because the performance of the system is greatly affected by the dividing method. For example, in Fig. 4–2(a), the search space is divided into four parts and each of them are assigned to processor p_1 to p_4. Here, we assume that p_2 found that the hatched area in the figure can be eliminated from the search space. Then p_2 must transmit that information to other processors. On the other hand, if we divide the search space as shown in Fig. 4–2(b), then p_2 can reduce the search space without communicating with other processors. Therefore, it is better to divide the search space so that the reduction can be done locally in a processor.

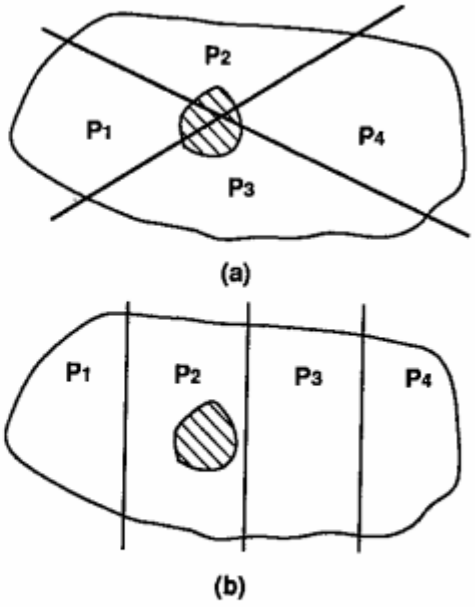

Fig. 4–2 Search Space Division

In the presumption problem, the search space has a tree structure. Each node in the search tree corresponds

to a possible presumption tree. Moreover, for a internal node of the search tree, each of its child node corresponds to a presumption tree which has longer description length than the parent node's corresponding one. Therefore, for example, the root node of the search tree corresponds to the simplest presumption tree.

If a search process examined node T, and the pruning condition for a child node of T is satisfied, then the subtree below the child node can be pruned (Fig. 4–3(a)). This means that the pruned area is included in a subtree which has node T as a root. In other word, parallel search for multiple disjoint subtrees can be performed independently. The algorithm we propose divides the search tree into several disjoint subtrees and searches each of them with individual processor (Fig. 4–3(b)).

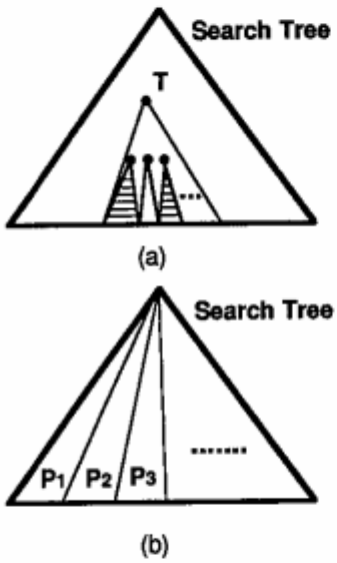

Fig. 4–3 Local Pruning

4.2 Global Pruning Mechanism

There is another kind of search tree pruning mechanism. If a certain process finds that a presumption tree T_0 has less description length than ever known, then each processor need not to test a tree that seems to have longer description length than T_0. The rest of this section describes details of this technique.

Here we consider two presumption trees T and T' such that $T' \succ T$. Then

$$\begin{aligned} & L1(T') + L2(T') \\ & \geq L1(T') + L2_{MIN} \\ & > L1(T) + L2_{MIN}. \end{aligned}$$

Here, if newly found tree T_0, which has shorter description length than ever known, satisfies the pruning condition :

$$L1(T') + L2_{MIN} \geq L(T_0)$$

then, from the above inequalities, we can conclude:

$$L1(T') + L2(T') > L(T_0)$$

$$\text{i.e. } L(T') > L(T_0).$$

Therefore, it is not necessary to examine T'. Therefore if we find a presumption tree which has shorter length than ever known, then some portion of the search space will be able to be eliminated.

However, reducible part of the search tree may be distributed widely throughout the search space. In other words, the pruning information should be announced to all of the other processors. Therefore, it is important to consider the trade-off between the increase of communication cost and the reduction of computation cost. That is, in a searching process, if a presumption tree is found to have shorter length than ever known, then the length of the tree should not always be announced to the other processors.

In order to solve the problem, a simple mechanism is incorporated. That is, the newly found length is transmitted only if it is over x bits smaller than the previously known least length. Here, x is a threshold value.

5 Implementation and Results

The learning algorithm was implemented in KL1 language on Multi-PSI, a distributed-memory multi processor machine. First, we implemented the algorithm with the local pruning mechanism. The experiments were performed by using up to 16 PEs in parallel. As a sample data, a fault history which comprised about 100 fault examples was given. The computation time was measured 5 times, and we took the average. The speedup curve of the example is shown in Fig. 5–1.

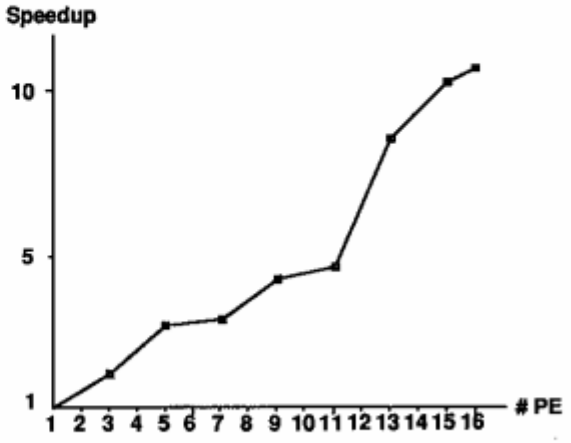

Fig. 5–1 Speedup of the Algorithm

The implementation using 16 PEs is about 11 times as fast as the sequential implementation (1 PE). There is a

possibility of further speedup by equalizing the load of each PE. An example of the overall load distribution is illustrated in Fig. 5–2. The difference of the load among the PEs may be improved by adding a dynamic load balancing mechanism into the system. Development of this mechanism is under investigation.

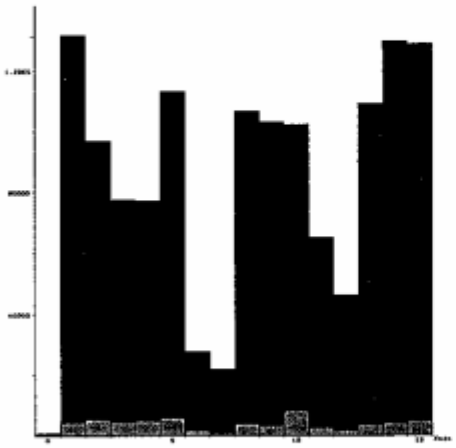

Fig. 5–2 Load Distribution

Next, we implemented the global pruning technique in addition to the local pruning mechanism. The threshold value for transmission is set to 2. This value was acquired empirically.

The performance of the algorithm with both the local and global pruning mechanism is shown in Table 5–1.

Table 5–1 Performance of the Algorithms

(a)

	No. of Reductions	
	Local	Local+Global
Example 1	870716	558661
(ratio)	1.00	0.64
Example 2	3602255	2588851
(ratio)	1.00	0.71
Example 3	30773602	23342853
(ratio)	1.00	0.76

(b)

	Execution Time (msec)	
	Local	Local+Global
Example 1	4522	3378
(ratio)	1.00	0.75
Example 2	16050	11282
(ratio)	1.00	0.70
Example 3	109892	89549
(ratio)	1.00	0.81

Each experiment is performed with three randomly generated examples. The number of reductions and the execution time are measured for the two versions of the algorithm. One is an algorithm with local pruning mechanism (Local), and another version incorporates both local and global pruning mechanism (Local+Global). Both of them are executed with 16 PEs.

The results show that the global pruning mechanism improved both of the number of reductions and execution time about 20% to 30% in comparison with the local pruning version. Fig. 5–3 shows an example of acquired presumption tree.

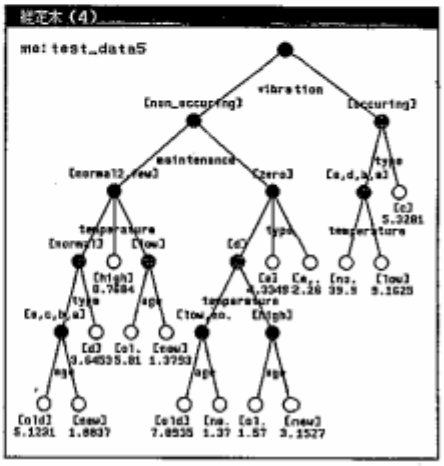

Fig. 5–3 Example of Acquired Tree

6 Conclusion

This paper has described a parallel learning algorithm for adaptive model-based diagnosis. The algorithm is based on branch-and-bound technique, and local and global pruning mechanisms are incorporated into the algorithm. The 16 PE implementation with local pruning mechanism is shown to be about 11 times as fast as the sequential one. Moreover, the global pruning mechanism is shown to have an ability to accelerate the parallel search.

Future work is to improve the heuristics used in the pruning process. If we can find more effective global pruning criterion which can be computed with low time complexity, it seems to be possible to perform superlinearly.

Acknowledgment

This research has been carried out as a part of the Fifth Generation Computer Project. The authors would like to thank Katsumi Nitta of the Institute for New Generation Computer Technology for his support, and to express their appreciation to Masahiro Yamamoto and Takeshi Yoshimura of NEC for their encouragement in this work.

References

[Davis 1984] Davis, R., "Diagnostic reasoning based on structure and behavior," *Artificial Intelligence,* Vol. 24, pp. 347–410, 1984.

[de Kleer 1987] de Kleer, J. and Williams, B. C., "Diagnosing multiple faults," *Artificial Intelligence,* Vol. 32, pp. 97–130, 1987.

[de Kleer 1989] de Kleer, J. and Williams, B. C., "Diagnosis with behavioral modes," *Proc. IJCAI-89,* Vol. 2, pp. 1324–1330, 1989.

[Genesereth 1984] Genesereth, M. R., "The use of design descriptions in automated diagnosis," *Artificial Intelligence,* Vol. 24, pp. 411–436, 1984.

[Koseki et al. 1990a] Koseki, Y., Nakakuki, Y., and Tanaka, M., "An adaptive model-Based diagnostic system," *Proc. PRICAI'90,* Vol. 1, pp. 104–109, 1990.

[Koseki et al. 1990b] Koseki, Y., Nakakuki, Y., and Tanaka, M., "An adaptive model-based diagnostic system and its learning method," *Proc. 4th Annual Conference of Japanese Society for Artificial Intelligence (in Japanese),* pp. 503–506 1990.

[Koseki 1989] Koseki, Y., "Experience learning in model-based diagnostic systems," *Proc. IJCAI-89,* Vol. 2, pp. 1356–1361, 1989.

[Nakakuki et al. 1990] Nakakuki, Y., Koseki, Y., and Tanaka, M., "Inductive learning in probabilistic domain," *Proc. AAAI-90,* Vol. 2, pp. 809–814, 1990.

[Nakakuki et al. 1991a] Nakakuki, Y., Koseki, Y., and Tanaka, M., "An adaptive model-based diagnostic strategy," *Proc. 5th Annual Conference of Japanese Society for Artificial Intelligence (in Japanese),* 1991.

[Nakakuki et al. 1991b] Nakakuki, Y., Koseki, Y., and Tanaka, M., "An inductive learning method and its application to diagnostic systems," *IPSJ SUG Reports 91-AI-74 (in Japanese),* Vol. 91 (3), pp. 19–28, 1991.

[Nakakuki et al. 1991c] Nakakuki, Y., Koseki, Y., and Tanaka, M., "Inductive learning of probabilistic knowledge," *Proc. 42nd Annual Convention IPS Japan (in Japanese),* 1991.

[Nakakuki et al. 1991d] Nakakuki, Y., Koseki, Y., and Tanaka, M., "A parallel algorithm for learning presumption tree," *Proc. 43rd Annual Convention IPS Japan (in Japanese),* 1991.

[Ohta et al. 1991a] Ohta, Y. et al., "A parallel processing method for an adaptive model-based diagnostic system," *Proc. 5th Annual Conference of Japanese Society for Artificial Intelligence (in Japanese),* 1991.

[Ohta et al. 1991b] Ohta, Y. et al., "A parallel processing method for an model-based diagnosis," *Proc. KL1 Programming Workshop (in Japanese),,* 1991.

[Quinlan 1986] Quinlan, J. R., "Induction of decision trees," *Machine Learning,* Vol. 1 (1), pp. 81–106, 1986.

[Rissanen 1978] Rissanen, J., "Modeling by shortest data description," *Automatica,* Vol. 14, pp. 465–471, 1978.

[Rissanen 1983] Rissanen, J., "A universal prior for integers and estimation by minimum description length," *Ann. of Statist.,* Vol. 11, pp. 416–431, 1983.

[Rissanen 1986] Rissanen, J., "Complexity of stings in the class of Markov sources," *IEEE Trans. on Information Theory,* Vol. 32 (4), pp. 526–531, 1986.

[Shortliffe 1976] Shortliffe, E. J., *Computer Based Medical Consultations: ba1MYCIN,* American Elsevier, New York 1976.

[Tanaka et al. 1989] Tanaka, M., Koseki, Y., and Nakakuki, Y., "A method of narrowing down suspects using experiential knowledge in model-based diagnostic systems," In *Proc. 39th Annual Convention IPS Japan (in Japanese),* pp. 243–244 1989.