An Efficient Technique for Design of ABFT Systems Based on Modified PD Graph

Choon-Sik Park and Mineo Kaneko
School of Information Science
Japan Advanced Institute of Science and Technology
1-1 Asahidai, Tatsunokuchi, Ishikawa 923-1292, Japan
{cspark,mkaneko}@jaist.ac.jp

Abstract

We present an efficient design method of Algorithm-Based Fault Tolerant(ABFT) systems based on Modified Processor-Data(MPD) graph which maintains the data dependent information between data elements for a given computation. We investigate the fault detectability and locatability for k-fault in terms of error patterns, and a check scheme for single-fault locating and two-fault detecting ABFT systems is derived. With the full use of data dependent information together with an appropriate error propagation model, the number of checks can be reduced compared with conventional processor-data model.

1. Introduction

The advent of cost-effective VLSI components in the past few years has made commercial development of multiprocessor systems feasible. Since the probability of one or more processors failing in such multiprocessor systems is quite large, it is desirable to build some fault tolerant feature into them. But the requirements for high performance and fault tolerance are seemingly contradictory. Cost effectiveness has always been a major concern in designing VLSI systems. Therefore a useful objective of research is to devise techniques for incorporating fault tolerance at lower cost without sacrificing the performance.

The technique called algorithm-based fault tolerance(ABFT), suggested by Huang and Abraham[2], deals with concurrent error checking at a high level to achieve the above objectives. ABFT techniques have been proposed for various signal processing computations such as matrix operation, Fast Fourier Transforms(FFT), QR Factorization, and so on. Due to the increasing popularity of ABFT applications it is desirable to have a general model and methodology for system analysis to design more efficient fault tolerant systems.

The first attempt analyzing ABFT systems was made by Banerjee and Abraham[2] who proposed a graph-theoretic model. Also, the matrix-based model presented in [7] simplified the analysis procedure by introducing the new necessary and sufficient conditions for the fault detectability and locatability of ABFT systems. These models have not addressed error generation and error propagation for a given system. Therefore there may exist some redundant error patterns in analyzing and designing ABFT systems.

Such redundant error patterns may be a cause of an increase in the complexity and the number of checks in analyzing and designing ABFT systems, respectively. This situation motivates the investigation of more efficient model so that the analysis procedure is simplified and the number of checks is reduced. To achieve such objectives, we introduce data dependent information between computation results computed by processors. Since data dependency gives an useful information for error generation/propagation, we can simplify the analysis procedure and reduce the number of checks by using data dependent information in analyzing and designing ABFT systems.

In this paper, we will discuss an analysis model based on MPD graph, its effectiveness in designing ABFT systems and constructing checks for k-fault detectable and locatable ABFT systems. Also, we will discuss *single-fault* detectability and locatability so that checks can be directly obtained from MPD graph, and provide a heuristic method in constructing checks for *single-fault* locatable and *two-fault* detectable ABFT systems.

The rest of the paper is organized as follows. In section 2 we define faults, errors, and checking operations with regard to ABFT systems. The analysis model based on MPD graph and its effectiveness are described in detail in Section 3. In section 4 we present the fault detectability and locatability for k-fault from the relations of error patterns and data elements. An efficient design of *single-fault* detecting and locating ABFT system from MPD graph is presented in section 5. Finally, we conclude in section 6.

2. Terminology

In this section, we will define faults, errors, and checking operations with regard to multiprocessor systems which are candidate architectures for the application of ABFT techniques. The basic terminologies are based on [7].

A fault is any condition that causes a malfunction in processor(s). An error is any discrepancy between the expected result of an operation and the actual result of the operation under fault of the processor which performs the operation. A fault in a processor is assumed to be manifested as an error in one or more data elements affected by it. In general, a fault in ABFT scheme is detected by detecting errors in the results generated by the processor. The problem of detection of various faults is translated to the problem of detecting errors in the computed results. However, we must note that certain types of faults may not produce any error at all. Therefore, if a particular fault does not produce any error in data elements computed by a processor, we say the fault to be $unobservable$ and may disregard the presence of that fault. That is, if a fault is unobservable, then we don't distinguish between this case and the fault-free case because the result of the computation is correct. Hence, an error implies the presence of a fault in the computing system.

The main target application regarding in this paper is the linear algebra based computations such as matrix operations and signal processing. And we assume throughout this paper that, whichever a processor is normal or faulty, the processor always outputs an erroneous result if at least one input data for computing the result is erroneous.

A collection of one or more faults(errors) is called a $fault(error)\ pattern$. If there are M processors in the system, there are $2^M - 1$ possible fault patterns. Fault patterns consisting of k or fewer elements(faulty processors) are called *k-fault*.

A $check$ on the data element is any combination of hardware and software procedures performed on the data elements to generate an output either 1 or 0. The set of data elements checked by a check is called its *data set*. We assume that the capability of a check is limited to a (g,h) check, and the (g,h) check is defined on g data elements such that the result of the checking operation is as follows:

1. The check reports error($i.e.$ outputs a 1) if it determines that there is at most h data elements in its data set being in error.
2. The check passes($i.e.$ outputs a 0) if it determines that there are no errors in the data elements in its data set.
3. The check is unpredictable if the number of erroneous elements in its data set is greater than h.

The checks introduce redundancies in the computation to detect and locate faults in the system. In general, the redundancy to implement the checks increases as h is greater. The (g,1) check should be more practically implemented, and we will use in this paper only (g,1) check with arbitrary but finite g.

We assume that the checks in a system consist of $c_1, c_2, \cdots, c_q, \cdots, c_Q$. The state of the checks in the system can be represented by a Q-bit binary sequence. The qth bit in the sequence is 1 if check c_q outputs 1, else it is 0. This Q-bit binary sequence is usually called the $syndrome$.

A system is said to *k-fault detectable* if some check reports error for any fault pattern of size at most k. Also a system is said to *k-fault locatable* if each fault pattern of size at most k can be uniquely identified by the syndrome. However, since an ABFT system indirectly tests the faults by errors and there may exist some unobservable fault, we will redefine the fault detectability and locatability in terms of error patterns for disregarding the existence of the unobservable faults.

Definition 1 (k-Fault Detectable) *An ABFT system is said to k-fault detectable if for each error pattern induced by k-fault, there is at least one check that certainly outputs 1.*

Definition 2 (k-Fault Locatable) *An ABFT system is said to k-fault locatable if for any pair of error patterns, one is induced by a fault pattern in k-fault and the other is induced by any other fault pattern in k-fault, there is some check that certainly gives a different output.*

3. Analysis Model using MPD Graph

3.1. MPD Graph Model

An algorithm which can be represented by *Dependence Graph*(DG), $G_D(V_D,E_D)$, is considered, where V_D denotes the set of vertices each of which represents an operation associated with data element, primary input or primary output, and E_D denotes the set of directed edges each of which represents the data dependency from source to destination vertices.

In the practical implementation, vertices in DG are mapped into processors by a mapping function M_f, where M_f is assumed to map vertices of primary inputs and primary outputs into themselves. As a result, an implementation of a given $G_D(V_D,E_D)$ is modeled as the computation graph $G_c(V_c,E_c)$, where V_c=$\{M_f(v)|v \in V_D\}$ is a set of vertices which represent processors, primary inputs and primary outputs and E_c=$\{(M_f(u),M_f(v))|(u,v)\in E_D, M_f(u)\neq M_f(v)\}$ is a set of directed edges which represent data dependencies between processors and between processors and primary inputs/outputs. Various implementations can be obtained by choosing various mapping functions. The problem of how to choose M_f is out of our concern, and we assume that the mapping function M_f is a priori given.

We assume that a given algorithm is executed on a set of M processors $p_1, p_2, \cdots, p_M$ and generates N data elements $d_1, d_2, \cdots, d_N$ as results. From G_c, we can obtain a directed acyclic graph $G(V,E)$ which is called MPD graph. $V(=V_p \cup V_d)$ denotes the set of nodes of processors(V_p) and data elements(V_d) and $E(=E_{pd} \cup E_{dd})$ denotes a set of edges each of which represents either the relation between processors and data elements, *i.e.*, if a processor p_m produces a data element d_n, then $(p_m, d_n) \in E_{pd}$, or data dependency between data elements, *i.e.*, if one data element d_i is used as an input for computing other data element d_j, then $(d_i, d_j) \in E_{dd}$.

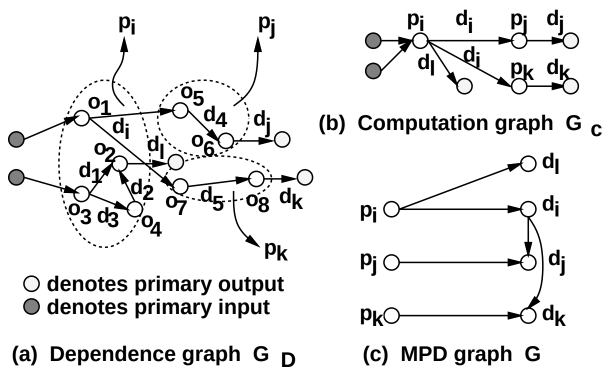

Figure 1. Description of MPD graph model.

In the following, we introduce several notations for fault patterns and error patterns. f_i^l is a set of i processors and indicates one fault pattern of the size i.

$$F_i = \bigcup_{l=1}^{C(M,i)} \{f_i^l\} \tag{1}$$

is the set of all fault patterns of their size exactly i, where $C(M,i)=\frac{M!}{(M-i)!i!}$. And

$$F^k = \bigcup_{i=1}^{k} F_i \tag{2}$$

is the set of all k-fault patterns.

On the other hand, e_i^{lj} is a subset of V_d and indicates one error pattern.

$$e_i^l = \{e_i^{l1}, e_i^{l2}, \cdots, e_i^{lj_l}\} \tag{3}$$

is the set of all error patterns induced by a fault pattern f_i^l.

$$E_i = \bigcup_{l=1}^{C(M,i)} e_i^l \tag{4}$$

$$E^k = \bigcup_{i=1}^{k} E_i \tag{5}$$

are the sets of error patterns induced by fault patterns in F_i and those in F^k, respectively.

To construct error patterns, if it is needed, we consider *adjacency matrices* PD and DD and a *reachable matrix* R as follows:

1. $[PD]_{ij}$: 1 if there is a directed edge from p_i to d_j and 0 otherwise.
2. $[DD]_{ij}$: 1 if there is a directed edge from d_i to d_j and 0 otherwise.
3. $[R]_{ij}$: 1 if there is a path from d_i to d_j and 0 otherwise.

From these two adjacency matrices and the reachable matrix, first we construct E_1. And then we obtain $E_2, E_3, \cdots, E_k$ from E_1, recursively. That is, E_i consists all of the available *pairwise unions* of all elements in E_1 and those in E_{i-1}.

3.2. Effect of MPD Model

To show the effectiveness of MPD model, we will consider the following algorithm as an example. Data element d_i is computed by operation o_i for $i = 1, 2, \cdots, M$. And data element d_i is used as an input for computing the data element d_{i+1} for $i = 1, 2, \cdots, M-1$ and d_M is not used as inputs for computing any other data elements. We assume that operations are mapped into processors in *one-to-one* fashion so that data element d_i is computed by processor p_i. The conventional PD graph[2] and the proposed MPD graph for this situation are illustrated in Figure 2.

Assume that we want to construct a set of checks such that the designated system is *single-fault detectable*. For the case of the conventional PD graph, M checks are needed to detect *single-fault* because a fault in p_1 may affect all of data elements to be erroneous, so all available error patterns induced by faulty processor p_1 are all of the available combinations of data elements $d_1, d_2, \cdots, d_M$. However, for the case of the proposed MPD graph which introduces data dependent information, we can detect single-fault to just one check for d_M because error pattern induced by faulty processor p_i always includes the data element d_M. Also there does not exist any check to locate single-fault in the conventional PD graph because the faulty processor

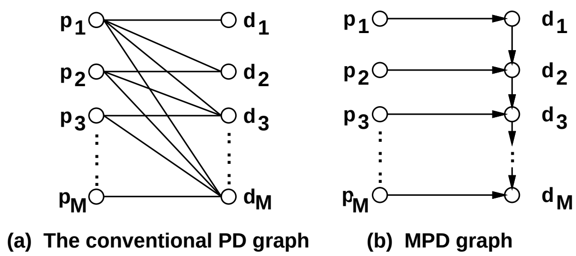

Figure 2. The conventional PD graph and MPD graph for a given algorithm.

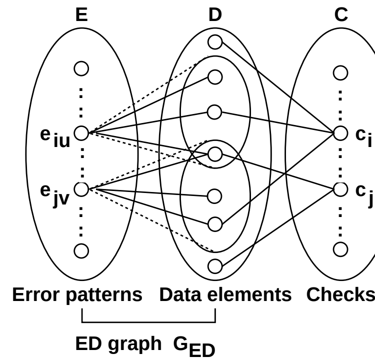

Figure 3. Construction of checks for k-fault detectable ABFT system.

p_1 and another faulty processors can not be identified by the $syndrome$. But single-fault in MPD graph can be located by M checks: one check for each data element d_i. Thus, there always exist checks to locate single-fault in the proposed MPD graph. In most cases that there exist data dependencies between data elements computed by processors, the designated ABFT system in the proposed MPD graph can be implemented with fewer checks than the conventional PD graph.

The details on fault detection and location based on MPD graph for ABFT system will be discussed in the following section.

4. Fault Detectability and Locatability in terms of Error Patterns

In this section, we describe fault detectable and locatable system based on the proposed MPD graph model. To simplify the notations, let f_i be a fault pattern in F^k and let $e_{iu}(\in\ E^k)$ be the u-th error pattern induced by a fault pattern f_i. To describe fault detectability and locatability, first we introduce an *undirected bipartite* graph $G_{ED}(V_{ED}, E_{ED})$ which describes the relation between error patterns and data elements. The set of vertices V_{ED}(=$E^k \cup V_d$) denotes the set of error patterns(E^k) and the set of data elements(V_d), and the set of edges E_{ED} denotes the relation between error patterns and data elements : if the data element d_n is contained in an error pattern e_{iu} induced by a fault pattern f_i, then there exists an undirected edge(e_{iu},d_n).

In the following, let $C = \{c_1, c_2, \cdots, c_q, \cdots, c_Q\}$ be a set of checks and $\langle(V,E),C\rangle$ be an ABFT system.

Lemma 1 *An ABFT system $\langle(V,E),C\rangle$ is k-fault detectable if and only if for each error pattern e_{iu}, $1 \le i \le |F^k|$, there is at least one check $c \in C$ such that* $|c \cap e_{iu}| = 1$.

Proof: (1) *sufficient condition*: If $|c_q \cap e_{iu}|$=1, then the check c_q certainly outputs 1 for an error pattern e_{iu} because exactly one data element in c_q is in error. Hence, there is at least one check which certainly outputs 1 for each error pattern induced by f_i, $1 \le i \le |F^k|$. (2) *necessary condition*: The proof is by contradiction. Suppose that, for a certain error pattern e_{iu}, $^\forall c_q$, [$|c_q \cap e_{iu}|$=0 or $|c_q \cap e_{iu}|$ $\ge$2]. If $|c_q \cap e_{iu}|$=0, then c_q certainly outputs 0 for the error pattern e_{iu}. If $|c_q \cap e_{iu}|$ $\ge$2, then the check c_q is unpredictable because the check c_q is (g,1) check. Hence, there is no check that can certainly output 1 for e_{iu}. This is a contradiction.

Now, we will discuss k-fault locatable system. Analyzing ABFT system for its fault locatability is a much harder problem when compared to the problem of analyzing the fault detectability. This is the reason that, in the case of fault locatability, we have to determine not only whether a fault pattern is detectable but also whether the fault pattern is distinguishable from other fault patterns.

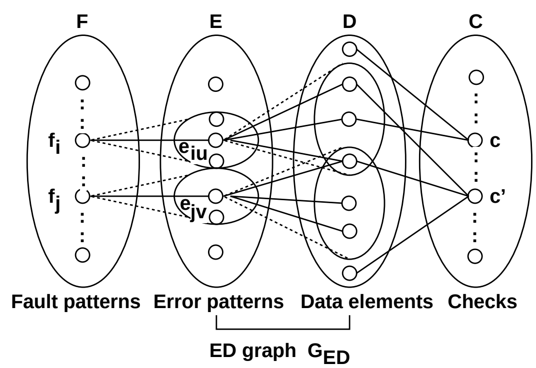

Figure 4. Construction of checks for k-fault locatable ABFT system.

Lemma 2 *If an ABFT system $\langle(V,E),C\rangle$ is k-fault locat-*

able, then for any pair of error patterns e_{iu} *and* e_{jv}, $i \neq j$, $1 \leq i \leq |F^k|$, $1 \leq j \leq |F^k|$, $e_{iu} \oplus e_{jv} \neq \emptyset$, *where* $\oplus$ *denotes the symmetric difference.*

Proof: The proof is by contradiction. Suppose that for $i \neq j$, the symmetric difference of the error patterns e_{iu} and e_{jv} is empty. When f_i and f_j induce e_{iu} and e_{jv}, respectively, the syndrome for f_i and the one for f_j are the same because $e_{iu} = e_{jv}$. Hence, there is no way that the checks can distinguish f_i and f_j for $i \neq j$. This is a contradiction.

Lemma 3 *An ABFT system* $\langle (V, E), C \rangle$ *is k-fault locatable if for each pair of error patterns* e_{iu} *and* e_{jv}, $i \neq j$, $1 \leq i \leq |F^k|$, $1 \leq j \leq |F^k|$, *there is at least one check-pair* $c \in C$ *and* $c' \in C$ *such that*

$$|c \cap (e_{iu} \oplus e_{jv})| = 1 \text{ and } |c \cap (e_{iu} \cap e_{jv})| = 0$$
$$|c' \cap e_{jv}| = 1 \text{ if } |c \cap (e_{iu} - e_{jv})| = 1$$
$$|c' \cap e_{iu}| = 1 \text{ if } |c \cap (e_{jv} - e_{iu})| = 1.$$

Proof: From the definition of k-fault locatability, the syndromes for two error patterns e_{iu} and e_{jv} which are induced by two-distinct fault patterns f_i and f_j, respectively, have to be different. Let d_{ij} be a data element contained in both $e_{iu} \oplus e_{jv}$ and c so that $|c \cap (e_{iu} \oplus e_{jv})|$=1 and $|c \cap (e_{iu} \cap e_{jv})|$=0. If d_{ij} is in e_{jv}, then $|c' \cap e_{iu}|$=1 and the partial syndrome cc' is 01 and 11(or 1X) for f_i and f_j, respectively, where X denotes that the check is unpredictable. Similarly, if d_{ij} is in e_{iu}, then $|c' \cap e_{jv}|$=1 and the partial syndrome cc' is 11(or 1X) and 01 for f_i and f_j, respectively. Therefore, f_i and f_j are detected and distinguished by the check-pair c and c'.

Lemma 4 *If an ABFT system* $\langle (V, E), C \rangle$ *is k-fault locatable due to Lemma 3, then it is also 2k-fault detectable.*

Proof: It is clear that if an ABFT system $\langle (V, E), C \rangle$ is k-fault locatable, then it is also k-fault detectable because from the proof of Lemma 3, the check-pair c and c' can detect e_{iu} and e_{jv}. The error patterns induced by $2k$-fault are obtained by taking all available pairwise unions of elements in E^k and elements in E_k. Hence we will prove whether the checks detect the error pattern $e_{iu} \cup e_{jv}$, $1 \leq i \leq |F^k|$, $1 \leq j \leq |F^k|$. Since exactly one data element in $e_{iu} \cup e_{jv}$ in E^{2k} is in data set of the check c, the error pattern $e_{iu} \cup e_{jv}$ can be detected by c from Lemma 1. Therefore, the checks which consist of these check-pairs can detect $2k$-fault.

5. Single-Fault Detecting and Locating System from MPD Graph

In the previous section, fault detectability and locatability are discussed in terms of error patterns. However, in general, to generate error patterns is a costly task, and also the number of error patterns is too large to maintain for analysis and synthesis. In this section, we describe *single-fault* detectability and locatability on MPD graph.

For a given MPD graph $G(V, E)$ with M processor nodes and N data nodes, $D(p_m)$ denotes the set of all data elements which are reachable from the processor p_m. Note that D(p_i) is equivalent to the largest error pattern induced by single-fault pattern $\{p_i\}$.

Let D_i=$\{D_{i1}, D_{i2}, \cdots, D_{iW_i}\}$, $i = 1, 2, \cdots, M$, be a subset family of D(p_i), where D(p_i)=$\bigcup_{w=1}^{W_i} D_{iw}$. D_{iw} denotes the set of all data elements which are reachable from the wth adjacent data element of the processor p_i. W_i is the number of adjacent data elements of p_i

Lemma 5 *An ABFT system* $\langle (V, E), C \rangle$ *is single-fault detectable if for each* D_i, $1 \leq i \leq M$, *there is a set of checks* $C_i = \{c_0, c_1, \cdots, c_s, \cdots, c_{S_i}\} \subseteq C$ *such that* C_i *is recursively(until* D_i^s *is empty) defined as follows,*

$$D_i^0 = D_i$$
$$\left| c_s \cap \left(\bigcup_{D_{iw} \in D_i^s} D_{iw} \right) \right| = 1$$
$$D_i^{s+1} = D_i^s - \bigcup_{|c_s \cap D_{iw}|=1} \{D_{iw}\}$$

Proof: Note that all of the available unions of elements in D_i become all error patterns for the single-fault pattern $\{p_i\}$. The check c_0 detects the largest error pattern(D(p_i)) and some other error patterns which are all of the available unions of D_{iw}'s such that $|c_0 \cap D_{iw}|$=1. The check c_s detects error patterns which are all of the available unions of D_{iw}'s such that $|c_s \cap D_{iw}|$=1. Together with the definition of D_i^{s+1}, $D_i - D_i^{s+1} = \bigcup_{\alpha=1}^{s} \left[\bigcup_{|c_\alpha \cap D_{iw}|=1} \{D_{iw}\} \right]$ and $\{c_0, c_1, \cdots, c_s\}$ can detect error patterns which are all of the available unions of elements in $D_i - D_i^{s+1}$. Since c_{s+1} is selected so that $\left| c_s \cap \left(\bigcup_{D_{iw} \in D_i^{s+1}} D_{iw} \right) \right| = 1$, then $|D_i - D_i^{s+2}| \geq |D_i - D_i^{s+1}| + 1$, and hence $D_i - D_i^{s+1}$ becomes to contain all elements in D_i after appropriate iterations.

Lemma 6 *An ABFT system* $\langle (V, E), C \rangle$ *is single-fault locatable if for each pair of* D_i *and* D_j, $i \neq j$, $1 \leq i \leq M$, $1 \leq j \leq M$, *there is a set of checks* C_{ij}=$C_r \cup (\bigcup_{r=0}^{R} C'_r) \subseteq C$, *where* C_r=$\{c_0, c_1, \cdots, c_r, \cdots, c_R\}$ *and* C'_r=$\{c'_{r0}, c'_{r1}, \cdots, c'_{rb}, \cdots, c'_{rB_r}\}$ *such that they are recursively(until either* D_i^r *or* D_j^r *is empty) defined as follows,*

$$D_i^0 = D_i$$
$$D_j^0 = D_j$$
$$\left| c_r \cap \left(\bigcup_{D_{iw} \in D_i^r} D_{iw} \oplus \bigcup_{D_{jz} \in D_j^r} D_{jz} \right) \right| = 1 \text{ and}$$

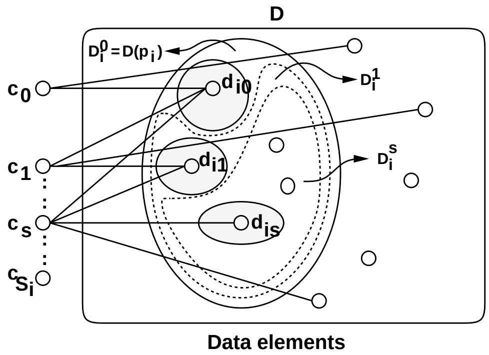

Figure 5. Construction of checks for single-fault detection.

$$\left|c_r \cap \left(\bigcup_{D_{iw}\in D_i^r} D_{iw} \cap \bigcup_{D_{jz}\in D_j^r} D_{jz}\right)\right| = 0$$

if $\left|c_r \cap \left(\bigcup_{D_{iw}\in D_i^r} D_{iw}\right)\right| = 1$, *then* C'_r, D_i^{r+1} *and* D_j^{r+1} *are,*

$$\left[\begin{array}{l} D_j^{r0} = D_j^r \\ \left|c'_{rb} \cap \left(\bigcup_{D_{jz}\in D_j^{rb}} D_{jz}\right)\right| = 1 \\ D_j^{r(b+1)} = D_j^{rb} - \bigcup_{|c'_{rb}\cap D_{jz}|=1} \{D_{jz}\} \\ D_i^{r+1} = D_i^r - \bigcup_{|c_r\cap D_{iw}|=1} \{D_{iw}\} \\ D_j^{r+1} = D_j^r \end{array}\right]$$

if $\left|c_r \cap \left(\bigcup_{D_{jz}\in D_j^r} D_{jz}\right)\right| = 1$, *then* C'_r, D_i^{r+1} *and* D_j^{r+1} *are,*

$$\left[\begin{array}{l} D_i^{r0} = D_i^r \\ \left|c'_{rb} \cap \left(\bigcup_{D_{iw}\in D_i^{rb}} D_{iw}\right)\right| = 1 \\ D_i^{r(b+1)} = D_i^{rb} - \bigcup_{|c'_{rb}\cap D_{iw}|=1} \{D_{iw}\} \\ D_i^{r+1} = D_i^r \\ D_j^{r+1} = D_j^r - \bigcup_{|c_r\cap D_{jz}|=1} \{D_{jz}\} \end{array}\right]$$

Proof: From Lemma 5, if $\left|c_r \cap \left(\bigcup_{D_{iw}\in D_i^r} D_{iw}\right)\right|$=1, then c_r detects error patterns which are all of the available unions of D_{iw}'s such that $|c_r \cap D_{iw}|$=1 and C'_r detects all error patterns in $\bigcup_{D_{jz}\in D_j^r} D_{jz}$, while c_r outputs 0 for every error pattern in $\bigcup_{D_{jz}\in D_j^r} D_{jz}$. Accordingly, the partial syndrome $c_r c'_{r0} c'_{r1} \cdots c'_{rB_r}$ is different for any one of error patterns which are all of the available unions of D_{iw}'s such that $|c_r \cap D_{iw}|$=1 and any one of error patterns in $\bigcup_{D_{jz}\in D_j^r} D_{jz}$. Now the remained pairs of error patterns to be distinguished are all element pairs of $D_i^r - \bigcup_{|c_r\cap D_{iw}|=1} \{D_{iw}\} = D_i^{r+1}$ and $D_j^r = D_j^{r+1}$. The case $\left|c_r \cap \left(\bigcup_{D_{jz}\in D_j^r} D_{jz}\right)\right|$=1 is the same but D_i^r and D_j^r are updated differently. Totally, the set of checks C_{ij} can distinguish fault patterns $\{p_i\}$ and $\{p_j\}$. Therefore, if there is a set of checks C_{ij} for $i \neq j$, $1 \leq i \leq M$, $1 \leq j \leq M$, then the ABFT system $\langle (V,E), C\rangle$ is single-fault locatable.

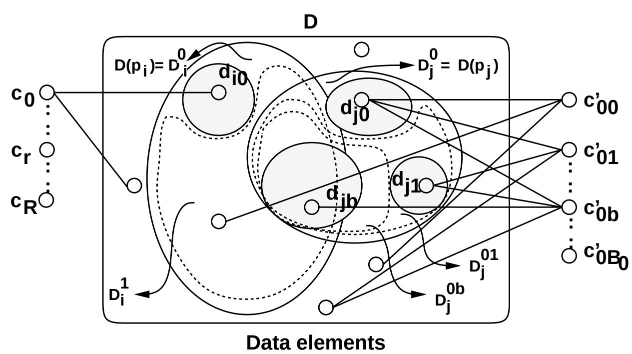

Figure 6. Construction of checks for single-fault location.

Lemma 7 *If an ABFT system $\langle (V,E), C\rangle$ is single-fault locatable due to Lemma 6, then it is also two-fault detectable.*

Proof: Let U_{ix} be one of all available unions of elements in D_i and let U_{jy} be one of all available unions of elements in D_j. Then U_{ij}(=$U_{ix} \cup U_{jy}$) is one of the error patterns induced by the fault pattern $\{p_i, p_j\}$. From Lemma 6, since exactly one data element in U_{ij} is in the data set of c_r, U_{ij} can be detected by c_r. Since every error pattern for the fault pattern $\{p_i, p_j\}$ has the form $U_{ix} \cup U_{jy}$, the single-fault locatable ABFT system by Lemma 6 is also two-fault detectable system.

Now, we introduce a greedy algorithm to construct checks for single-fault locating and two-fault detecting ABFT system. Let c and c' be a check-pair satisfying Lemma 6: c and c' are in C_r and C'_r, respectively. Let D_{d_n} be the set of D_{iw}'s(or D_{jz}'s) which includes the data element d_n. The algorithm shown in Figure 7 finds data elements to be checked by checks, and always returns CHK if there exists a set of checks given by Lemma 6.

6. Conclusions

In this paper, we present a model based on MPD graph which is addressed the data dependent information in designing ABFT systems. By using this model, the number of checks can be reduced for most cases that exist data dependencies between results computed by processors. Also we present k-fault detectability and locatability in terms of error patterns, and present single-fault detectability and locatability on MPD graph. Finally, we introduced a greedy algorithm for single-fault locating and two-fault detecting ABFT system.

Acknowledgments

This work is supported in part by the Ministry of ESSC under Grant-in-aid for Scientific Research No.09450158, Japan and by the Research Body CAD21, Tokyo Institute of Technology, Japan.

References

[1] B.Vinnakota and N.K.Jha. Synthesis of algorithm-based fault tolerant systems from dependence graphs. *IEEE Trans. on Parallel and Distributed Systems*, 4(8), August 1993.

[2] K.H.Huang and J.A.Abraham. Algorithm-based fault tolerance for matrix operations. *IEEE Trans. on comput.*, 33(6):518–528, June 1984.

[3] P.Banejee and J.A.Abraham. Concurrent fault diagnosis in multiple processor systems. *Proc. 16th Symp. Fault Toleramt Comput.*, pages 298–303, June 1986.

[4] P.Banerjee and J.A.Abraham. Bounds on algorithm-based fault tolerance in multiple processor systems. *IEEE Trans. on comput.*, 35(4):296–306, April 1986.

[5] R.Sitarman and N.K.Jha. Optimal design of checks for error detection and location in fault tolerant multiprocessor systems. *Proc. Int. Symp. Fault Tolerant Comput.*, pages 396–406, 1992.

[6] S.Yajnik and N.K.Jha. Graceful degradation in algorithm-based fault tolerant multiprocessor systems. *IEEE Trans. on Parallel and Distributed Systems*, 8(2):137–153, February 1997.

[7] V.S.S.Nair, J.A.Abraham, and P.Banerjee. Efficient techniques for the analysis of algorithm-based fault tolerant schemes. *IEEE Trans. on comput.*, 45(4), April 1996.

CHK_LOCATE($D(p_m)$,D_m,CHK)

$CHK \leftarrow \emptyset$
for $1 \leq i < j \leq M$
  $Q_i \leftarrow D(p_i)$
  $Q_j \leftarrow D(p_j)$
  $T_i \leftarrow D_i$
  $T_j \leftarrow D_j$
  while$(Q_i \neq \emptyset$ & $Q_j \neq \emptyset)$
    if(c *is not in* CHK *such that* $|c \cap (Q_i \oplus Q_j)| = 1$)
      $c \leftarrow d_n$ *with the maximum cardinality* $|D_{d_n} \cap (Q_i \oplus Q_j)|$ *in* $Q_i \oplus Q_j$
    $CHK \leftarrow CHK \cup \{c\}$
    if$(|c \cap Q_i| = 1)$
      $Q \leftarrow Q_j$
      $T \leftarrow T_j$
      while$(Q \neq \emptyset)$
        if(c' *is not in* CHK *such that* $|c' \cap Q| = 1$)
          $c' \leftarrow d_n$ *with the maximum cardinality* $|D_{d_n}|$ *in* Q
        $CHK \leftarrow CHK \cup \{c'\}$
        $T \leftarrow T - \bigcup_{|c' \cap D_{jz}|=1} \{D_{jz}\}$
        $Q \leftarrow \bigcup_{D_{jz} \in T} D_{jz}$
      $T_i \leftarrow T_i - \bigcup_{|c \cap D_{iw}|=1} \{D_{iw}\}$
      $Q_i \leftarrow \bigcup_{D_{iw} \in T_i} D_{iw}$
    else
      $Q \leftarrow Q_i$
      $T \leftarrow T_i$
      while$(Q \neq \emptyset)$
        if(c' *is not in* CHK *such that* $|c' \cap Q| = 1$)
          $c' \leftarrow d_n$ *with the maximum cardinality* $|D_{d_n}|$ *in* Q
        $CHK \leftarrow CHK \cup \{c'\}$
        $T \leftarrow T - \bigcup_{|c' \cap D_{iw}|=1} \{D_{iw}\}$
        $Q \leftarrow \bigcup_{D_{iw} \in T} D_{iw}$
      $T_j \leftarrow T_j - \bigcup_{|c \cap D_{jz}|=1} \{D_{jz}\}$
      $Q_j \leftarrow \bigcup_{D_{jz} \in T_j} D_{jz}$
return CHK

Figure 7. A greedy algorithm for constructing checks of single-fault locating and two-fault detecting ABFT systems.