Effective Approximate Fault Diagnosis of Systems with Inhomogeneous Test Invalidation

Tamás Bartha
Technical University of Budapest
Department of Measurement and Instrument Engineering
Mûegyetem rkp. 9, H-1521 Budapest, Hungary
bartha@mmt.bme.hu

Abstract

System-level fault diagnosis is a methodology to identify the failed components in a multiprocessor system. The traditional approach to system-level diagnosis does not take into consideration many important aspects of modern multiprocessor architectures. This paper examines a special class of multiprocessors, called massively parallel computers. As a practical example, the Parsytec GCel system is presented. The paper describes a new method developed for the Parsytec GCel, called local information diagnosis. The diagnostic algorithm is based on the generalized test invalidation model, therefore it is applicable to a wide range of systems, including inhomogeneous ones. Due to the employed syndrome decoding mechanism, the space and computational complexity of the algorithm is also smaller than in conventional methods.

1. Introduction

The price to performance ratio of commercial microprocessors rapidly decreases. Still, many computation-intensive scientific and technical applications require significantly more processing power than the capabilities of single processor systems. Parallel processing is an architectural solution to this problem. In *massively parallel computers* (MPCs) several thousand processing elements provide the necessary computing capacity. Although modern electronic components built in these systems have a very low permanent fault rate, their large number and the long processing time result in reduced overall dependability. Errors occurring during normal operation must be tolerated, eliminating their effect on the system. This can be achieved by a *fault-tolerant* architecture.

Automatic fault diagnosis is an important part of constructing reliable hardware. Its task is to locate the failed components in the system. In large multiprocessors a circuit-level fault diagnosis is impractical, therefore *system-level diagnosis* considers only processor faults. Processors periodically execute tests on each other. If an error was detected, the diagnostic procedure identifies faulty units by analyzing the collection of the test results (called the *syndrome*). Once these units were identified, they are logically isolated and the system is reconfigured to continue the error-free operation.

1.1. Generalized test invalidation

Fault-free tester units are assumed to test other units correctly (tests are assumed to be *complete*). Still, faulty testers may produce and distribute incorrect test results that do not reflect the real fault state of the tested unit. In other words, errors affect the validity of test outcomes. This effect, modelled by *test invalidation*, makes the decoding of the syndrome more difficult. The two most frequently used test invalidation models are the *symmetric* or PMC model [2], and the *asymmetric* or BGM model[3]:

– the symmetric model has a very pessimistic view about the faulty units: test results by such units are arbitrary, independent on the state of the tested unit.
– the asymmetric model states, that two units fail identically with a negligible probability. Therefore a faulty unit tests another faulty unit differently from the symmetric model: the test always fails.

Generalized test invalidation describes practical test invalidation mechanisms in a unified framework. Consequently, it allows algorithms to deal with inhomogeneous systems consisting of different units and test invalidation. A complete description and analysis of generalized test invalidation is covered by [1]. The model can be outlined by Table 1.

In the model, results generated by a faulty tester unit may take three values: always pass, always fail or pass/fail in a non-deterministic way. These results are also denoted by the constants 0, 1 and X. Test invalidation models are characterized by the actual *C* and *D* values, so there are nine possible models. The symmetric or PMC model has T_{XX} invalidation, and the asymmetric or BGM model has T_{X1} invalidation according to the generalized model.

Tester unit	Unit under test	Test result
fault-free	fault-free	0
fault-free	faulty	1
faulty	fault-free	$C \in \{0, 1, X\}$
faulty	faulty	$D \in \{0, 1, X\}$

Table 1. Generalized test invalidation

The relationship between a tester and a tested unit is defined by *one-step implication rules*. The existence of a one-step implication depends on three factors:

- the test invalidation of both units,
- the fault state of both units, and
- the actual test outcome.

One-step implication rules have four different types: tautology, forward implication, backward implication, and contradiction. Note, that of these four only contradiction rules provide sure inferences (as they exclude the incorrect fault state from diagnosis), provided that the initial model is valid [10].

Units can be classified by deriving implication chains starting from a supposed fault state of a unit. A possible diagnosis is a contradiction-free classification of every unit. One-step implications are combined into implication chains using *transitivity*. The collection of all valid inferences included in the syndrome is the result of *transitive closure*. Thus, the diagnostic procedure is equivalent to the state-space search known from artificial intelligence. Unfortunately, in most cases there are multiple candidate diagnoses, due to diagnostic uncertainty. The aim of one-step system-level diagnosis is to select from this set the one corresponding to the actual fault configuration. The selection is possible if further restriction can be made on the fault model (like a diagnostic *t*-limit on the number of faulty units) [11, 12].

2. Modelled system

The primary target of the diagnosis algorithm presented in this paper was the Parsytec GCel massively parallel reliable multiprocessor (see Figure 1).

The processing elements are Inmos T805 transputers. Sixteen transputers (plus one spare) are grouped together in one *cluster*. Clusters are the basic building blocks of the machine. Four of them constitute a so-called *GigaCube*. A GigaCube forms a physical unit: it has its own temperature and voltage monitoring facilities, and a *control node*. They can be arranged into a $4 \times 4 \times 8$ spatial array, so in its full configuration the system is scalable up to 16384 transputers. Each of the transputers is connected to the others via four data links, providing fast (up to 20 MBit/s) serial interprocessor communication. These point-to-point connections are structured in a two-dimensional mesh topology, called the *Data Network*.

Control nodes are additional Inmos T805 transputers, dedicated to the supervision and control of system operation. They play an important part in fault tolerance: they periodically test all the processing elements in the clusters of the GigaCube, and replace the faulty transputer with the spare. Control nodes are also responsible for booting, job control, and dynamic configuration management. They communicate over a separate interconnection network called the *Control Network*.

Peripheral I/O management is done by a stand-alone host machine (usually a Sun workstation) connected to the

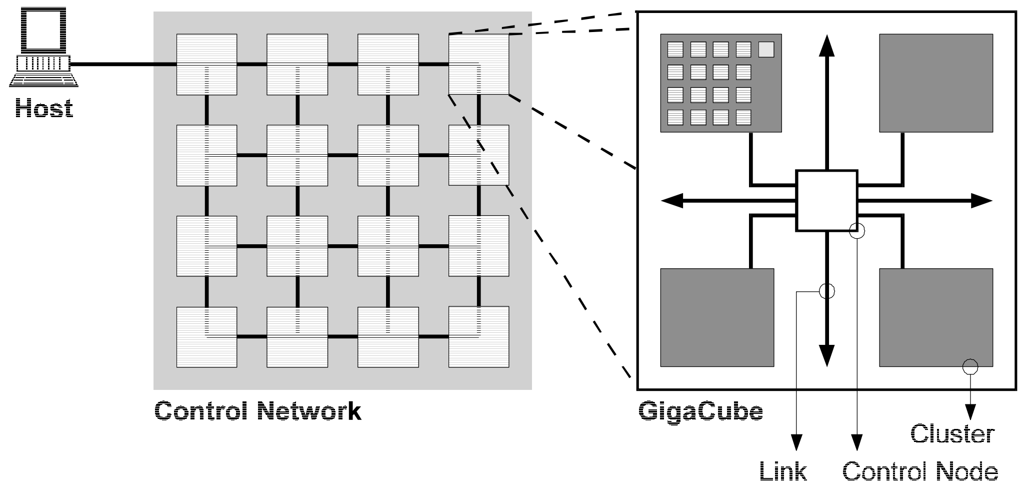

Figure 1. Structural layout of the Parsytec GCel

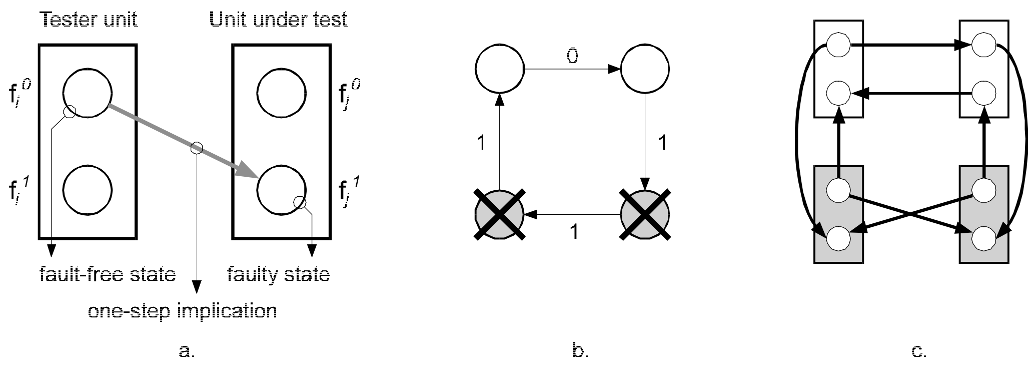

Figure 2. (a) Elements of the implication graph model, (b) Testing graph (PMC test invalidation), (c) Corresponding implication graph

Parsytec GCel machine. However, peripherals are also allowed to be connected directly to the processing elements.

3. Diagnostic model

In this paper the same assumptions are used on the fault model, as in the traditional approach to system-level diagnosis:

- the system consists of *intelligent units* able to individually and completely test each other,
- testing is possible only via normal communication,
- the test structure is predefined,
- the test invalidation can be inhomogeneous, but it conforms to the generalized model,
- faults are permanent,
- faults in communication links are not considered,
- the diagnosis is centralized.

The predefined testing assignment is described by a digraph $T = (U, E)$, called the *testing graph*. Here, U represents the set of units in the system, and E corresponds to the set of tests. Thus, a directed edge $t_{ij} \in E$ exists, iff unit u_i tests unit u_j. The syndrome is a set of $a_{ij} \in \{0, 1\}$ test results, where 0 indicates a passed, 1 a failed test. Test results are shown as labels on the edges. Unit u_i is a *tester* of unit u_j if $t_{ij} \in E$, or a *tested unit* if $t_{ji} \in E$. Testers of u_i are referred to as $\Gamma^{-1}(u_i)$, the set of units directly tested by u_i as $\Gamma(u_i)$. The collection of tester and tested units is the set of *neighbors*: $\vartheta(u_i) = \Gamma^{-1}(u_i) \cup \Gamma(u_i)$.

The diagnostic inferences can be expressed in a digraph $I = (U, F, P)$, called the *implication graph* (see *Figure 2*). The graph is composed of the set of units U in the system, the set of fault states F, and the set of one-step implications P derived from the generalized test invalidation model. These elements are represented by boxes, nodes and directed edges, respectively. Each unit $u_i = \{f_i^0, f_i^1\}$ is composed of two fault states: f_i^0 symbolizes the fault-free, f_i^1 the faulty state. A directed edge connects two nodes if a one-step implication exists between the corresponding fault states of the given units. No distinction is made between forward and backward rules. Loop edges representing tautology are not included in the implication graph (see Figure 2).

A diagnostic algorithm must partition the nodes in F into two disjunct sets: the set of diagnosed states D and the set of invalid states R. The cardinality of $|D \cap u_i| = 1$, $|R \cap u_i| = 1$ for every $u_i \in U$. The fault states in D constitute the diagnosis result. The following properties of the implication graph can help this process:

– implication chains appear in the graph as edge sequences,
– a contradiction exists iff there is an edge sequence between the two states of the same unit,
– when a contradiction is detected, the fault state of every unit being a part of the implication chain leading to the contradiction can be *surely* inferred,
– for two nodes u_i, $u_j \in U$ without sure classification, the exists a $p_{ij} \in P$ implication iff both of them belong the same set, either D or R (*consistency*),
– if a unit is not involved in any contradictions, its fault state can be surely classified iff there is additional information available on the fault model.

[1] presented an algorithm for system-level diagnosis under the generalized test invalidation model. The *Selényi* algorithm works with any diagnosable system topology. The repeated applications of one-step implications (i.e. the transitive closure) are estimated by the logical closure of the M adjacency matrix of the implication graph. Sure classification is obtained by the contradictions found in the closed M^* matrix. The fault state of the remaining unclassified units is identified on the basis of a t-limit.

4. Motivation of approximate diagnosis

Existing system-level diagnosis algorithms attempt to *completely* and *correctly* diagnose the whole system after one round of testing. To achieve this goal, generally they require some of the following restrictions on the fault model [5]:

- a particular type of interconnection topology,
- homogeneous test invalidation,
- predetermined *t*-limit on the number of faults.

Another important issue is the space and computational complexity of diagnosis. The diagnosis problem in its most general form is NP-complete. In the above restricted situations the algorithms typically have better, polynomial complexity. Yet, taking into consideration some practical system characteristics, even these acceptable results can be improved:

Local complexity. MPCs use regular interconnection topologies. Regular topologies have a uniform structure with a small and constant local complexity. After the occurrence of an error, its effects also appear locally.

System size. The traditional *t*-limit is independent on the size of the multiprocessor. Using it as a diagnosability measure gives very pessimistic and impractical results in such large systems as MPCs [8, 9].

Fault sets. The expected amount of faults in MPCs is low compared to the system size. Fault sets may exist in the following three configurations:

1. the faults are scattered in the system,
2. the faults are close to each other in one group,
3. a combination of the above two cases.

In the first case, the syndrome contains failed test results only at the faulty unit. These units are surrounded with fault-free testers. There is correct diagnostic information available on all units, so the syndrome is easy to decode, even if the number of faults exceeds the *t*-limit.

In the second case, units on the border of the fault group have fault-free testers, so they can be diagnosed correctly. The units inside the fault group, however, are separated from the fault-free units. Consequently, their fault state cannot be identified in the worst-case if the test invalidation model contains diagnostic uncertainty.

A diagnostic algorithm can exploit all of these issues by processing only the local diagnostic information. Furthermore, the space and computational complexity of such an approximate diagnostic algorithm is linear: $O(n)$. If the algorithm is executed in more iterations, the set of diagnostic inferences is extended by transitivity (see Section 5).

4.1. Existing algorithms

Since fault-free units always test other units correctly, a test outcome of value 1 indicates that either the tester or the tested unit is faulty (i.e. the test is error detecting). A simple algorithm can count the number of tests failed on a unit. Formally, the k_i sum of failed tests at unit u_i is computed as $k_i = \sum_{u_j \in \Gamma^{-1}(u_i)} a_{ji}$.

There are two existing ways of using k_i in diagnosis. The approach described by Fussel and Rangarajan compares k_i to a predefined limit [6]. If the sum is lower than the limit, the algorithm classifies unit u_i as fault-free. To provide asymptotically correct diagnosis, the algorithm conducts multiple tests on the same unit. On the other hand, the *Majority* algorithm by Blough et al. considers the k_i sum as the result of a majority vote for faulty classification [7]. The paper shows, that using this simple diagnosis strategy correct diagnosis is assured with high probability in a class of systems that includes hypercube architecture.

Both of the above methods take into consideration only that a fault-free tester tests its neighbor correctly. They do not take advantage of the additional information included in less restricted test invalidation models, neither exploit transitivity of implications. Therefore, their performance is particularly poor if the faulty units are located close to each other. A new algorithm which uses also the additional information included in the implication graph is presented in the next section.

5. Diagnosis using local information

In the first iteration of the diagnosis algorithm we can exploit the fact that certain units can be surely classified in one step. Sure decisions come from two sources: one-step contradiction rules included in the generalized model, and the inferences based on formal logic shown in Figure 4. (The two pair of inferences in one column give the same result, they can be derived from each other.) Implications drawn from surely diagnosed units result in new sure classifications. This way the amount of reliable information increases in each step, significantly improving the performance of the algorithm.

For unclassified units in the first iteration (this includes units that can be surely classified only by traversing the implication graph in multiple steps), an idea similar to the algorithm *Majority* can be applied. The more implications support a fault state, the higher the probability is that selecting it as the diagnosed state will not lead to inconsistency. So, the number of edges constituting the edge sequences ending in a given state can be employed as a measure for its suitability. Hence, a diagnosis algorithm may use the following classification strategy:

1. for every fault state f_i^0, f_i^1 in the implication graph: compute the number of edges in all the edge sequences leading to it,
2. from f_i^0, f_i^1 select the state which has a higher number of supporting implications. If the two numbers are equal, select the f_i^0 as the diagnosed state.

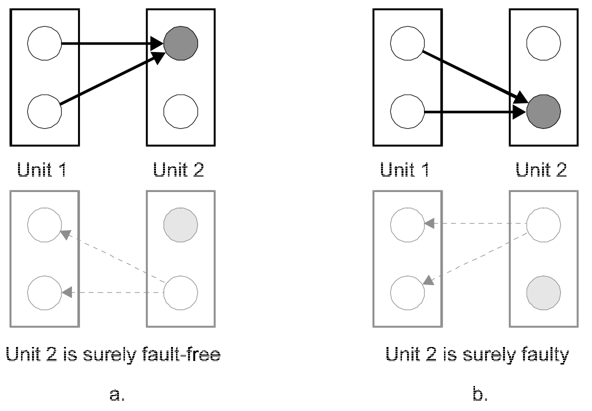

Figure 3. Sure implications based on formal logic

Lemma 1. *Although the algorithm using the above construction cannot identify contradictions, it will correctly diagnose the units having a sure classification.*

Proof: Since there is a contradiction at unit u_i, a directed path exists between the two states f_i^0 and f_i^1. Assume, that the path consists of q edges ($q \geq 1$), and leads from f_i^0 to f_i^1 (i.e. the fault state of unit u_i can surely be classified as faulty). Let the number of edges in paths leading to f_i^0 be equal to p. Then, the r number of edges in paths leading to f_i^1 is $r = p + q > p$. Thus, the algorithm will always correctly select f_i^1. •

Determining the number of edges in all the edge sequences leading to each node of the implication graph and generating the transitive closure have the same computational complexity. Instead of the actual number of implications supporting a fault state, we can estimate the suitability of the decisions by taking into account only the at most k length part of the implication chains (where $1 \leq k \leq n$). In the simplest case only direct edges from the neighbor units are counted. Essentially this is a majority vote of the neighbor units using all of the implications in the generalized test invalidation model. Counting paths of length 2 takes advantage of transitivity as well. Further extending the considered path length we get better estimations on the number of supporting units, thus

```
Algorithm CountInferencePaths
{ initialization }
for each $u_i \in V$ do
  $C_a^0[i] \leftarrow 0$, $C_p^0[i] \leftarrow 0$, $C_t^0[i] \leftarrow -1$
  $C_a^1[i] \leftarrow 0$, $C_p^1[i] \leftarrow 0$, $C_t^1[i] \leftarrow -1$
end for
{ compute the inferences leading to $f_i^0$ and $f_i^1$ }
for each iteration do
  for each $u_i \in U$ do
    for each $p_{jj} \in P$ do
      if $p_{jj}$: $f_i^0 \rightarrow f_j^0$ then
        $C_a^0[j] \leftarrow C_a^0[j] + (C_p^0[i] - C_t^0[i])*W[p_{ij}]$
      else if $p_{jj}$: $f_i^0 \rightarrow f_j^1$ then
        $C_a^1[j] \leftarrow C_a^1[j] + (C_p^0[i] - C_t^0[i])*W[p_{ij}]$
      else if $p_{jj}$: $f_i^1 \rightarrow f_j^0$ then
        $C_a^0[j] \leftarrow C_a^0[j] + (C_p^1[i] - C_t^1[i])*W[p_{ij}]$
      else if $p_{jj}$: $f_i^0 \rightarrow f_i^1$ then
        $C_a^1[j] \leftarrow C_a^1[j] + (C_p^1[i] - C_t^1[i])*W[p_{ij}]$
    end for
  end for
  for each $u_i \in U$ do
    $C_t^0[i] \leftarrow C_p^0[i]$, $C_p^0[i] \leftarrow C_a^0[i]$
    $C_t^1[i] \leftarrow C_p^1[i]$, $C_p^1[i] \leftarrow C_a^1[i]$
end for
```

Figure 4. Algorithm for counting the inferences leading to a fault state

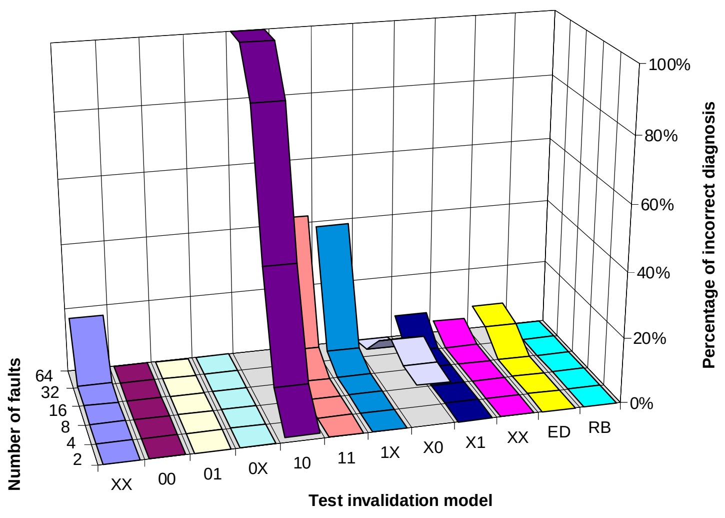

Figure 5. Percentage of diagnostic rounds with incorrect diagnosis in the function of faulty units

increasing the probability of consistent diagnosis.

Consistency is desirable in approximate diagnosis, but it will lead to false decisions if the multiple candidate diagnoses have equal significance. Additional information on the fault model can be included in the implication graph model by introducing a weight function *W* on the edges. The algorithm *CountPaths* (given in Figure 3) computes in $C_a^0[i]$ and $C_a^1[i]$ the weighted number of *l* length edge sequences ending in fault states f_i^0 and f_i^1 in the *l*th iteration. Its advantage that it is suitable for distributed implementation as well, since units obtain the necessary information only from their neighbors. The price for the simple and distributed structure is that the algorithm gives distorted results if the implication graph has loops. In spite of this, the diagnosis algorithm using *CountPaths* gave promising measurement results (see Section 5.1).

Correct identification of all faulty units in approximate diagnosis using only local information is not possible in every situation. Two types of incorrect diagnosis exist:

– *benign* misdiagnosis: some of the fault-free units are classified as faulty,
– *malign* misdiagnosis: some of the faulty units are classified as fault-free.

In an MPC benign misdiagnosis may be permitted. Provided that the number of misdiagnosed units is relatively low, surplus degradation will not significantly affect performance due to the large inherent redundancy. On the other hand, in many cases malign misdiagnosis is inadmissible, as unrecognized faulty units may distribute erroneous data corrupting the whole system.

As mentioned in Section 4, two fault configurations must be correctly identified: scattered faults, and the borders of fault groups. Units inside fault groups are inaccessible from fault-free units, they do not take part in the system operation. Their fault state is not diagnosable, so neither type of misdiagnoses on these units have significance.

5.1. Measurements

The diagnosis algorithm was measured using a two-dimensional torus topology, similar to the grid topology in the Parsytec GCel massively parallel computer. The torus contained 16×16 processing elements. Results were produced in a homogeneous system with every test invalidation model. Two inhomogeneous systems were also used: in the first random test invalidations were distributed evenly, in the other homogeneous test invalidation regions were placed randomly in it (these measurements are referred to as **ED** and **RB** in the figures). The edge weight function was chosen to produce good results with homogeneous symmetric invalidation. Four important properties of the algorithm were examined:

1. incorrect diagnosis in the function of faults,
2. incorrect diagnosis in the function of iterations,
3. diagnosis on the border of fault groups,
4. time complexity in the function of system size.

Figure 5 shows the percentage of diagnostic rounds with incorrect diagnosis in the function of faults. The number randomly injected faulty units was 2, 4, 8, 16, 32 and 64 (note, that the diagnostic *t*-limit for the employed

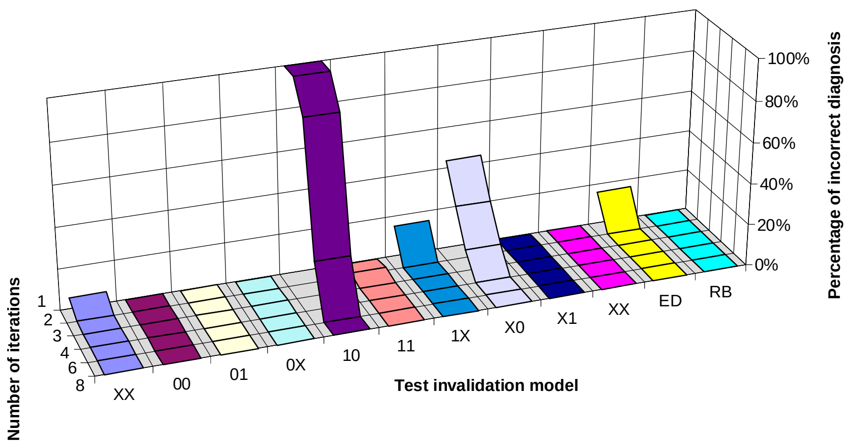

Figure 6. Percentage of diagnostic rounds with incorrect diagnosis in the function of iterations

topology is only 4). The algorithm was iterated 3 times in each diagnostic round. As the results indicate, in the majority of test invalidation models all units are diagnosed correctly even if 16 faulty units are present in the system.

The worst results were obtained in case of the T_{10} test invalidation model. This special fault model contains no diagnostic uncertainty. Here, a global method can be used to easily separate the units into two identically behaving sets. Then, assuming that the majority of units is fault-free, the set containing less elements can be surely classified as faulty. Yet, this simple model is very hard to handle in local information diagnosis.

The T_{1X} and T_{X0} models do not provide much more diagnostic information than the previous model. Although their results are better, the amount of incorrect diagnoses is still high. The syndrome in other models can always be correctly decoded even if 16 faults are in the system.

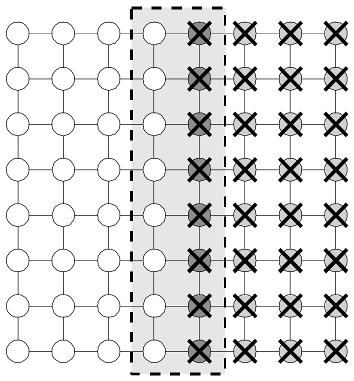

Figure 7. Fault pattern to examine diagnosis on the fault group borders

Figure 6 displays the effect of multiple iterations on diagnosis performance. In this measurement, 16 faults were injected randomly in the system. The number of iterations was 1, 2, 3, 4, 6, and 8. As shown on Figure 6, in more iterations transitivity and the additional knowledge represented by the edge weight function radically improve correctness at every test invalidation model.

The effect of fault group borders was examined using the fault pattern on Figure 7. In this experiment only units within the area surrounded by the dotted rectangle were checked for incorrect diagnosis. For the test invalidation models included in [4] (i.e. T_{00}, T_{01}, T_{0X}, T_{11}, T_{1X}, T_{X1}, T_{XX}) no unit on the fault group borders was misdiagnosed.

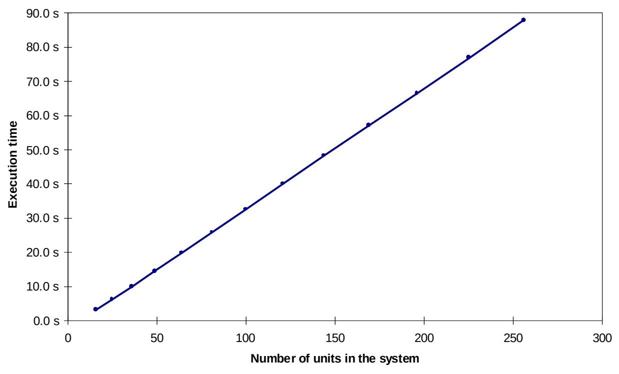

Figure 8. Time complexity in the function of system size

Figure 8 presents the total time of 1024 consequent executions of the algorithm in 3 iterations. The size of the system was increased from 4×4 units to 16×16 units. The number of faulty units was 8. The curve of measurement results is linear, justifying the theoretical results.

6. Conclusions

The paper has described a novel approach to efficient approximate diagnosis of massive parallel multiprocessor systems. The presented algorithm is based on the implication graph representation of the generalized test invalidation model. It uses a simple measure for the suitability of fault states, which expresses the consistency requirement of diagnosis. This measure is proven to correctly characterize units which have sure classification after the transitive closure. Additional information on the fault model is also included in the algorithm, in the form of an edge weight function. A simple method to compute the estimation of the above suitability measure was introduced. This way the new algorithm allows to scale the diagnosis between performance and accuracy. Measurement results were given to show the characteristics of the algorithm.

References

[1] E. Selényi, "Generalization of System-Level Diagnosis Theory," D.Sc. Thesis, Budapest, Technical University of Budapest, 1985.

[2] F. Preparata, G. Metze, and R.Chien, "On the Connection Assigment Problem of Diagnosable Systems," *IEEE Trans. on Computers*, vol. EC-16, no. 6, pp. 848-854, Dec. 1967.

[3] F. Barsi, F. Grandolini, and P. Maestrini, "A Theory of Diagnosability of Digital Systems," *IEEE Trans. on Computers*, vol. C-25, no. 6, pp. 585-593, June 1976.

[4] A. Friedman, and L. Simonchini, "System-Level Fault Diagnosis," *IEEE Computer*, pp. 47-53, March 1980.

[5] M. Barborak, M. Malek, and A. Dahbura, "The Consensus Problem in Fault-Tolerant Computing," *ACM Computing Surveys*, vol. 25, no. 2, June 1993.

[6] D. Fussel, S. Rangarajan, "Probabilistic Diagnosis of Multiprocessor Systems with Arbitrary Connectivity," *19th Int. IEEE Symp. on Fault-Tolerant Computing*, pp. 560-565, 1989.

[7] D. Blough, G. Sullivan, and G. Masson, "Fault Diagnosis for Sparsely Interconnected Multiprocessor Systems," *19th Int. IEEE Symp. on Fault-Tolerant Computing*, pp. 62-69, 1989.

[8] A. Somani, V. Agarwal, and D. Avis, "A Generalized Theory for System Level Diagnosis," *IEEE Trans. on Computers*, vol. C-36, no. 5, pp. 538-546, May 1987.

[9] A. Somani, and V. Agarwal, "Distributed Diagnosis Algorithms for Regular Interconnection Structures," *IEEE Trans. on Computers*, vol. 41, no. 7, pp. 899-906, July 1992.

[10] A. Pataricza, K. Tilly, E. Selényi, and M. Dal Cin, "A Constraint Based Approach to System-Level Diagnosis," Internal Report, Erlangen, Friedrich-Alexander University, 1994.

[11] J. Altmann, T. Bartha, A. Pataricza, "On Integrating Error Detection into a Fault Diagnosis Algorithm for Massively Parallel Computers," *IEEE Int. Symp. on Parallel and Distributed Systems*, pp. 154-164, 1995.

[12] J. Altmann, T. Bartha, A. Pataricza, "An Event-Driven Approach to Multiprocessor Diagnosis," *8th Symp. on Microcomputer and Microprocessor Applications*, pp. 109-118, 1994.