Annales UMCS Informatica AI 2 (2004) 143-151

Annales UMCS
Informatica
Lublin-Polonia
Sectio AI

http://www.annales.umcs.lublin.pl/

Algorithmical and topological methods of fault tolerance assurance

Mirosław Hajder, Paweł Dymora*
Department of Distributed Systems, Faculty of Electrical and Computer Engineering, Rzeszów University of Technology, Wincentego Pola 2, 35-959 Rzeszów

Abstract

Special interest in designing, constructing and exploitation of fault tolerant systems is connected with the need of attending to critical infrastructure systems. In this paper we focus on presentation and analysis of the requirements demanded on the context of fault tolerance assurance in the critical infrastructure systems. Especially, we take into consideration algorithmical and topological methods of fault tolerance assurance.

1. Introduction

In term of *fault tolerant systems* we describe these computer systems, among others, which despite faults in their hardware or software module still preserve their abilities to function in the whole or limited range. Problems of improper functioning includes the three main terms: *failure, fault and error. Failure* refers to an occurrence of the system of physical module damages. *Fault* is a condition existing in a hardware or software module that may lead to the failure of the module. *Error* is an incorrect response from hardware or software module caused by the existence of faults [1].

Faults may occur on different designing or exploitation levels. The highest level on which possible functioning faults may arise is the specification level when as a result of the designer's mistake during system designing an improper algorithmical, architectural or structural solutions were used. The next level on which possible faults may occur is the level of specifications realization. These faults are related with improper implementation of earlier worked out specification on hardware or software system modules. Another level of the faults occurrence is the production and exploitation level on which failure of one or a few system modules can take place mostly as a result of not obeying the specification and exploitation parameters. Obviously faults may also occur as a result of outer factors such as radiation, electromagnetic field or operator mistakes.

* Corresponding author: *e-mail address*: dymorap@prz.rzeszow.pl

From the activities point of view whose aim is to minimize the damaging effects to the system functioning we can distinguish three fundamentally different approaches. *Fault avoidance* is based on supplying the system with the elements allowing, on the basis of the functioning results, analysis and system modules diagnosis to counteract arising of potential functioning errors. *Fault masking* is based on excluding system elements which are the source of errors. *Fault tolerance* is the system ability to continue computations in the faulty system. Fault tolerance can take advantage of fault masking as well as system reconfiguration [2].

In our work we consider only the topology reliability. The main requirements for fault tolerant topology can be formulated as follows:

1. Topology should ensure the required level of fault tolerance without any excessive increase of its realization costs.
2. Topology should ensure the ability to decentralize the system diagnostic methods using relatively simple means.
3. Topology should ensure the ability of simple routing of sent information with the mineralized level of hardware support.

From the topological point of view the topology designing procedure reduces to the assurance of minimal product value of diameter and designed topology degree. In other words, parameters characterizing the topology fault tolerance are their survivability and cohesion.

In this paper we focus not so much on designing procedures of fault tolerant systems as on the methods and means of system reconfiguration in order to preserve qualitative and quantitative system characteristics.

2. The critical infrastructure systems, and fault tolerance

The limited level of survivability and cohesion characterizes all contemporary computer systems. It results from the fact that each of currently used system elements or programs may yield to a fault as a result of improper use, mistakes made in a designing procedure wear out, etc. The elements of computer structure are duplicated in time, hardware and information in order to eliminate or minimize the effects of the faults occurrence. In this way, fault tolerant structures come into being which in the case of faults occurrence makes that structure preserves its functions in the whole or limited range.

Among the computer systems we can distinguish a varied failure treatment. On the one hand, there exists systems in which limited realization of system functions for example in the efficiency range does not make using of its resources difficult. But there exists the whole group of systems in which determination of the efficiency parameters in particular limitation of system functions is unacceptable. The management systems of electric power networks, air or railway transportation are an example of such systems. Such systems are often called critical infrastructure systems. Special interest in designing,

constructing and exploitation of fault tolerant systems is connected with a need of attending to critical infrastructure systems [3,4].

As mentioned above the critical infrastructure system should be characterized by specific features which guarantees its fault tolerance in the case of possible failure. In particular in the range of executed functions the system architecture is a derivative of requirements demanded from them. Let us think over what features critical infrastructure system should be characterized by. The analysis is begun with the topological parameters.

The basic principle of the critical infrastructure system functioning is the maximal availability assurance of system resources. For this reason applying the shared medium topology is considered to be questionable. Certainly a better solution would be to apply the direct topologies which guarantee dedicated character of connects. In particular, in these systems point to point connections can be applied because of traffic routing requirement elimination. Such networks are not fault tolerant because it is impossible to set alternative connection links in it. The recommended feature of topological structures in the fault tolerant systems is preservation of all system topological characteristics besides a diameter. But let us notice that as a result of this system class complexity, especially of the significant number of nodes it is hardly possible to fulfill the above constraint. The topologies of critical infrastructure systems in principle are neither symmetric nor regular topologies. Besides, for of the security reasons they exclude the need of excessive topology development in order to make it symmetric. Other important features of the critical infrastructure systems are hardware and software heterogeneity and also data distribution. Heterogeneity is based on realization of varied functions by the nodes implemented with the varied architecture hardware thereby varied capabilities. The data distribution feature is related with the distributed data processing. Data are often gathered and shared with the other users directly from the place of their rise. Thus, it is necessary to ensure the efficient nodes co-operation in both the mutual data sharing process and its processing [4].

Mutual relationships between the selected features of the critical infrastructure systems are shown in Fig. 1.

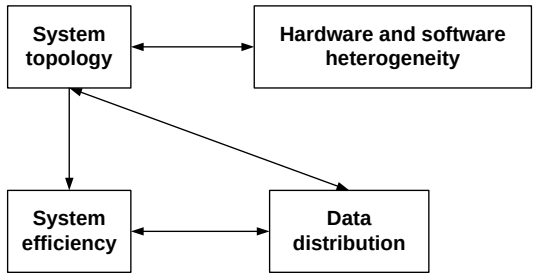

Fig. 1. Relationships between the features of the critical infrastructure systems

3. Reconfiguration as an instrument of preserving fault tolerance

Reconfiguration is one of basic methods of fault tolerance ensuring in the critical infrastructure systems. A special attention was arouse by reconfiguration algorithms in the context of distributed computation systems in which a significant computation power is achieved by connecting a lot of independent processing units. In most of these systems in order to ensure their fault tolerance the method is used which allows to apply additional processing units whose aim is to take over computation functions from the faulty node.

If we assume that in the case of occurrence of processing element failure the number of computed tasks in a system would be invariable, then it can turn out that critical resource of the system is an operating memory. Due to that fact the reconfiguration algorithms can be classified from the essential memory size point of view that is necessary to continue computation. On the one hand we can distinguish algorithms demanding invariableness of the whole system memory size but on the other hand, there exist algorithms which demand only memory size invariableness of faulty processing element or its neighbor processing elements. Other classification is based on specifying changes degree which occur directly after failure. In order to ensure the maximally fast system reconfiguration it is necessary to minimize the number of changes introduced to the system in case of failure occurrence. The above changes concern both the interconnection topology and tasks assigned to particular processing elements. In order to ensure the presented requirements usually the methods requiring only local reconfiguration are applied. In this case only faulty node and nodes directly neighboring with it would be subordinate to appropriate reconfiguration. But in practice such a method may turn out to be too expensive in application. Due to that fact in reconfigurations the idea which is based on the assumption that processing elements are gathered in groups to which spare elements are added is applied. To increase the universality of presented method it is assumed that spare elements would serve not only the group (cluster) for which they were assigned but also neighbor processor clusters. On the one hand, such a solution guarantees the increase of system reconfiguration speed. On the other hand, it causes the decrease of reconfiguration flexibility thereby limits operating reliability of the whole system.

In the light of presented considerations, the reconfiguration algorithms may be classified into two groups: local algorithms and global algorithms. In the case of local algorithms at the expense of redundancy the maximized reconfiguration speed is achieved. Global algorithms guarantee a redundancy optimization that is achieved at the expense of reconfiguration range and the realization time.

In order to realize the presented reconfiguration it is required to apply the system architecture with redundancy. We classify the system elements into primary elements and spare elements. In the case of primary element failure this element is replaced with the spare element.

Let us consider the example of system with cluster organization including the set of spare elements with hypercube interconnections.

It is assumed that the computation system is organized in the form of clusters. Each of them has a form of hypercube additionally equipped with redundant elements. Fig. 2 shows the structure in which except for 16 primary elements connected into four-dimensional hypercube, it uses 8 spare elements. The connection method of primary elements with the spare ones is presented in Fig. 2b.

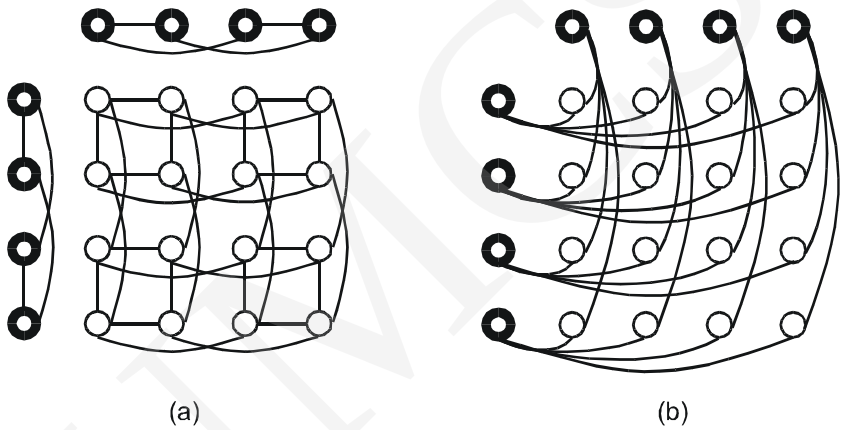

Fig. 2. Expanded four-dimensional hypercube

Each of the primary system elements may be denoted as (i, j), where $1 \le i, j \le 4$. The spare nodes may be denoted as $(0, j)$, where $1 \le j \le 4$ or $(i,0)$, $1 \le i \le 4$. In this way fault tolerant cluster – FTBB (*Fault Tolerant Basic Block*) was constructed. This FTBB cluster consists of 24 elements where each primary node (i, j) might be replaced with one of two spare nodes $(0, j)$ or $(i,0)$. Now, let us focus on reconfiguration realized in the FTBB cluster.

In the FTBB system there is a possibility of such a system failure that the whole system would be out of order. It takes place when none of the spare elements $(0, j)$ or $(i,0)$ would be able to replace the faulty node (i, j). Regardless of the system reconfiguration type (local, global) the faulty node must be replaced with the elements from the FTBB cluster. For example, if the node (i, j) fails in the first place one should try to replace such a node with the node $(i,0)$. If the node $(i,0)$ is unavailable it should be replaced (if possible) with the node $(0, j)$. The realization order of the replacing (reconfiguration) is free. But it is advisable to work out a certain replacing method. For example, if $i < j$ replacement might begin with the node $(i,0)$ or with the node $(0, j)$. This solution guarantees the optimization of both available spare element

organizations (column, row). Besides it is characterized by significant speed. In order to explain the presented reasoning let us consider the following example.

Let us assume that Fig. 3 shows structure where the following nodes (2,1), (2,2), (3,3), (1,2) and (4,2) failed. The failures followed the presented order. If the discussed above reconfiguration method is used then reconfiguration scheme would look like the presented one in Fig. 3a. It is not difficult to notice that four nodes in the case of new failure would not be able to be replaced with the spare elements. These nodes are represented in Fig. 3 by double circles. These nodes are (1,1), (1,3), (4,1) and (4,3).

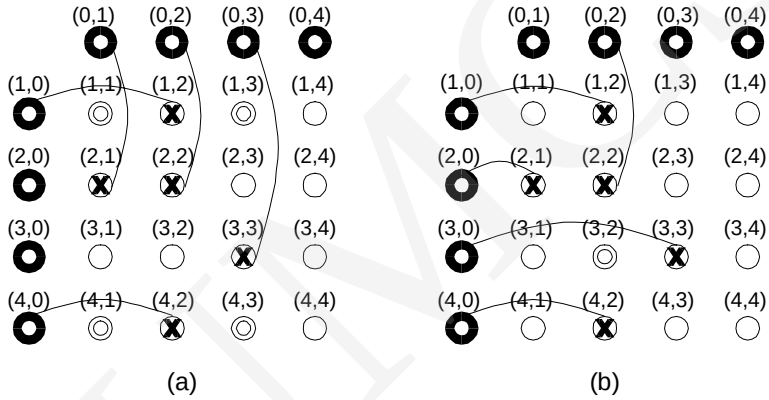

Fig. 3. Local (a) and global (b) reconfigurations

Applying the global reconfiguration may give the results shown in Fig. 3b. In this case only failure of the node (3,2) would not be repairable. Reconfiguration may also be used for other topologies like e.g. mesh, torus. It is also acceptable to use a much more modest set of redundant elements. Now, let us consider the case shown in Fig. 4.

Fig. 4a shows the graph G representing a distributed system based on the 3 x 3 torus topology $T_{3,3}$. For a subset Z of $V(G)$ let $G(Z)$ denote the subgraph induced in G by the nodes in Z. Let $N_G(u)$ denote the set of neighbors of a node u in G. For example in Fig. 4a, $N_G(9) = \{3,6,7,8\}$ where G is the 3 x 3 torus topology $T_{3,3}$. If F is the set of faulty nodes in G, then $G - F$ denotes the graph obtained by deleting the nodes and all edges incident on them from G. Fig. 4b shows the graph $T_{3,3} - \{5,9\}$ obtained when nodes 5 and 9 become faulty. We consider only the node failures. Faulty edges can be modeled as faults in the nodes on which these edges were incidental.

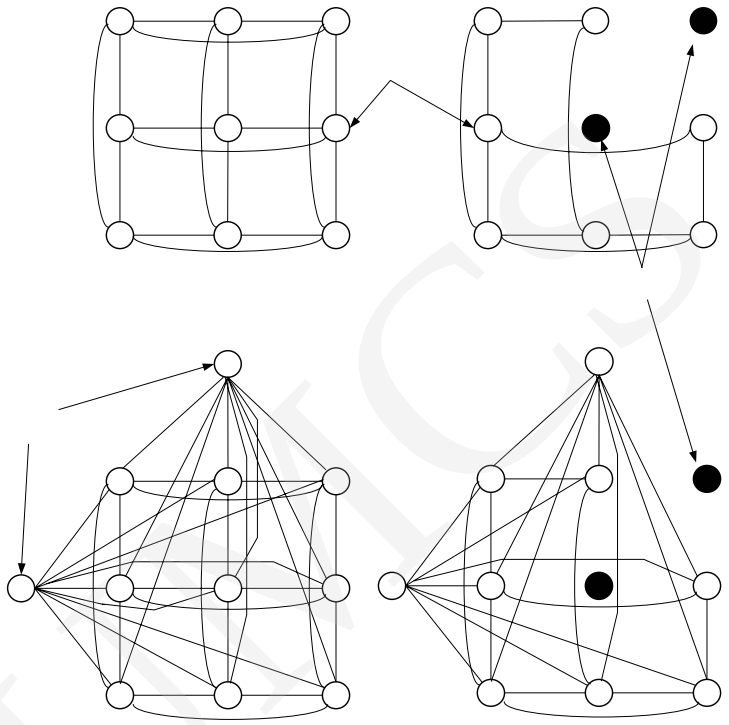

Fig. 4. Reconfiguration around faults in the torus topology with the restricted set of redundant elements

A supergraph $G'[k,G]$ of G is a k-fault tolerant realization of G, if for any set F of k nodes of G', $G'-F$ contain a subgraph isomorphic to G. Fig. 4c shows an example of a 2-fault tolerant supergraph $G'[2,T_{3,3}]$ of $T_{3,3}$ presented in Fig. 4a obtained by inserting the redundant nodes $r_1$and r_2, and the redundant links. The 2-fault tolerant supergraph of Fig. 4c is obtained by a construction in which the redundant nodes are connected to all the nodes of $T_{3,3}$ and to one another. The reconfiguration of this 2-fault tolerant supergraph around faulty nodes 5 and 9 is shown in Fig. 4d, where the redundant nodes r_1 and r_2 directly replace nodes 5 and 9, respectively.

An alternative to constructing k-fault tolerant supergraphs is to partition the node set of G into subsets $V_1,...,V_t$, introduce disjoint sets of k_i redundant nodes in each V_i to make the induced subgraph $G(V_i)$ the k_i-fault tolerant realization. Such a supergraph of G is denoted by $G'[\{k_1,k_2,...,k_t\},G]$ and is said to be a $\{k_1,k_2,...,k_t\}$ fault tolerant realization of G [5].

4. Reconfiguration algorithm

The basics of the reconfiguration process is the mapping of faulty nodes of G onto those fault free including those redundant of G'. The new node labels under this mapping determine a fault free copy of G. In such a mapping from $V(G)$ to $V(G')$ if node v is mapped to a node u the neighbors of v in G have to be mapped to a subset of the neighbors of u in G'. Such a mapping in G' is called a G-mapping. In the reconfiguratin method we designate a small subset Z_v of nodes of G' to which v can be mapped for reconfiguring around faults. Each node u in Z_v is called a node mapper of v denoted by $M_u(v)$ and Z_v the mapping set of v denoted by $mapp(v)$. A mapping graph $H(G)$ of a graph G is a directed graph whose node set consists of all the nodes of G and some redundant nodes, and whose edge set is $\{(u,v): v \in V(G), u \in mapp(v)\}$. A k-mapping graph is the one whose implied supergraph is k-fault tolerant. A mapping set for a node v of G in $H(G)$ is an ordered set $(v_1,..,v_h)$ of nodes in $V(H(G))$, such that $M_{v_1}(v)$, $M_{v_j}(v_{j-1})$, for $1 < j \le h$ and v_h is the only redundant node in this ordered set. A mapping set for a subset Z of $V(G)$ is a set of disjoint mapping sets for each node in Z, none of which contains any node of Z.

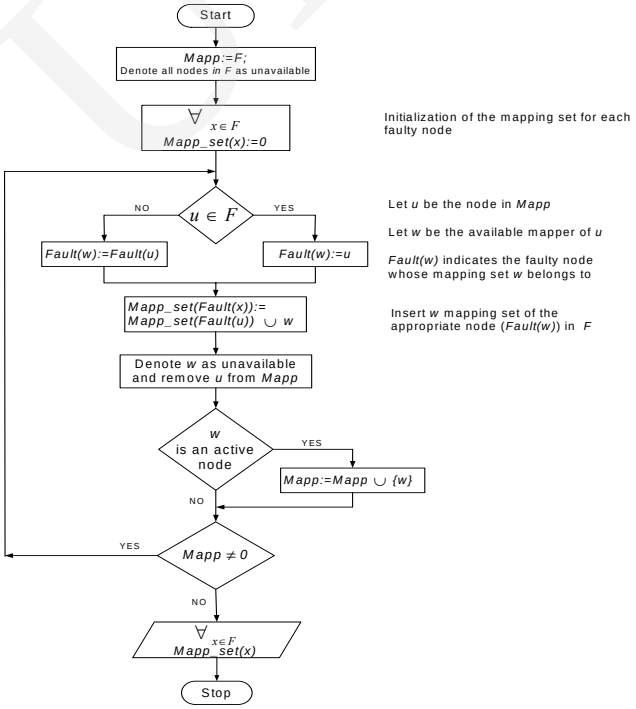

Fig. 5. An algorithm that finds mapping set for the fault set F in the mapping graph $H(G)$

The presented mapping set for v specifies a *G*-mapping which can be performed when v is faulty, so that v is mapped to v_1 which, in turn, is mapped to v_2, and so on, until v_{h-1} is mapped to the redundant node v_h, thus configuring it into the new nonfaulty system.

An algorithm that finds mapping set for *k* or fewer faulty nodes is presented in Fig. 5. This algorithm is based on the theorem which confirms that if each node of *G* is mapped exactly by *k*, the other nodes in the mapping graph $H(G)$ of *G*, and $H(G)$ are acyclic, then $H(G)$ is k-mapping graph [5,6].

5. Conclusions

In the fault tolerant systems its accessibility is mainly achieved by the topological parameters of the interconnection network optimization. Fault tolerance may be guaranteed by the reconfiguration of the existing resources so that the basic system functional parameters would not change. In such a case the system efficiency is possible to get worsen.

An alternative method is applying of the redundant elements which may be used in the reconfiguration process. The use of the large number of redundant elements causes the increase of the system fault tolerance without any system capacity degradation. However, such a situation results in a significant increase of the system realization costs. In the other case when the lower number of redundant elements is applied the system realization cost is lower but the system capacity degradation increases. Hence, applying of the optimal number of the redundant elements in the system reconfiguration process is an issue whose solution allows to realize structures with the increased level of fault tolerance by the relatively acceptable worsening of capacity parameters and increase of the costs [7].

The structure reconfiguration of the distributed critical infrastructure systems should be realized on the basis of the topologies characterized by the presented topological parameters especially regular and symmetric topologies.

References

[1] Hajder M., Loutskii H., Stręciwilk W., *Informatyka – Wirtualna podróż w świat systemów i sieci komputerowych*, Wydawnictwo WSIiZ, Rzeszów, (2002), in Polish.
[2] Pradhan D.K., Reddy S.M., *A fault tolerant communication architectures for distributed systems*, IEEE Transactions on Computers, (31)(9).
[3] Elder M.C., *Fault tolerance in critical information systems*, University of Virginia, (2001).
[4] Hajder M., Mazurek M., Dymora P., Hajder L., *Bezpieczeństwo systemów uszkadzalnych*, Zeszyty Naukowe WSIiZ w Rzeszowie, Rzeszów, (2001), in Polish.
[5] Dutt S., Hayes J.P., *Some practical issues in the design of fault tolerant multiprocessors*, IEEE Transactions on Computers, 31(5).
[6] Dolan A., Aldous J.M., *Introduction to Networks and Algoritms*, Wiley, (1993) 560.
[7] Hajder M., Mazurek M., Dymora P., *Virtual topologies of multinode wide networks*, Poznań University of Technology, Poznań, (2002), 197.