On Some Resources Placement Schemes in the 4-Dimensional Soft Degradable Hypercube Processors Network

Jan Chudzikiewicz and Zbigniew Zieliński

Military University of Technology, Faculty of Cybernetics,
ul. S. Kaliskiego 2, 00-908 Warszawa, Poland
{jchudzikiewicz,zzielinski}@wat.edu.pl

Abstract. In the paper some properties of the perfect resources placement in the 4-dimensional hypercube processors network with soft degradation are investigated. The two types of network processors (resource processors - I/O ports and working processors) are taken into consideration. In the work the notion of $(m,d|G)$ – perfect resources placement in a structure G of the hypercube type is extended to $(k|G)$ – perfect placement concept, that is a such allocation of k resources which minimizes the average distances between the working processors and resource processors in the structure G. Two algorithms for determining $(m,d|G)$ – perfect resources placement and the $(k|G)$ – resources perfect placement in a structure G of the hypercube network was given. The average number of working processors in the degraded 4-dimensional network with a given order (degree of degradation) for the $(1,1|G)$ resource placement is determined. This value characterizes the loss of the network computing capabilities resulting from the increase of the degree of network degradation.

Keywords: resources placement, hypercube network, fault-tolerant system.

1 Introduction

In a processors network, there may be resources, such as I/O processors or software components, that each processor needs to access. However, because of expense or frequency of use, resources in a network system might be shared to reduce the overall system cost. In general, then, the problem of resource placement is how should a limited number of copies of a resource be disseminated throughout a system giving comparable access to all processors. One of the situation to be considered in the context of resource allocation in fault-tolerant system is multiple-connection, in which every node without the resource is in connection with more than one copy of a resource. Multiple-connection gives rise to less contention, possibly yielding better performance, and reduces potential performance degradation after a copy is lost, because every node still can get access to one resource in the same number of hops.

W. Zamojski et al. (eds.), *Proceedings of the Ninth International Conference DepCoS-RELCOMEX*, Advances in Intelligent Systems and Computing 286,
DOI: 10.1007/978-3-319-07013-1_13, © Springer International Publishing Switzerland 2014

One of considered in the literature the resource placement problem is a combination of the distance-d and the m adjacency problems, where a non-resource node must be a distance of at most d from m resource nodes [1]-[3]. In [4] a perfect deployment has been introduced and analyzed which is defined as the minimum number of resources processors which are accessible by working processors. Each working processor has access to the at least m resources processors at a distance of not more than d. The definition of perfect distance-d placement of m copies of resources in the processors network was fairly commonly used in torus-type networks [3],[4]. Several resource placement techniques have been proposed for the hypercube or cube-type network [5]- [7], and different error correcting codes have been used to solve this problem.

An interconnection network with the hypercube logical structure is a well-known interconnection model for multiprocessor systems [8]-[10]. Such networks possess already numerous applications in critical systems and still they are the field of interest of many theoretical studies concerning system level diagnostics ([11]-[13]) or resource placement problem [6], [14]. More commonly, we could observe the usage processors networks in critical application areas [15] (military, aerospace or medical systems etc.). Such networks are (mostly) used in real-time systems, which require a very high data reliability processing throughout all the network life cycle.

We investigate the 4-dimensional hypercube processors network with two types of processors. The first type of processors are resource processors (i.e. I/O ports). These processors mediate in communication with other systems (sensors networks) and keep data obtained from them. The second type of processors are working processors. These processors have to get access to data obtained from sensors and are processing data from sensors. We also assume that sensors are not mobile. The execution of some tasks by a working processor requires an access to resources, also some results obtained by working processors must be submitted by a resource processors to other systems [14].

Further we assume that 4-dimensional hypercube is a soft degradable processors network [11]. In such kind of networks a processor identified as faulty is not repaired (or replaced) but access to it is blocked. New (degraded) network continues work under the condition that it meets special requirements [14],[16], which may involve possibility of applying of some resources placement schema (e.g. processors with connected I/O ports). In the work [16] an analysis of the different patterns of reconfiguration in networks with soft degradation was conducted, where these schemes were divided into software and hardware ones. Analyzed in this paper the way of the hypercube network reconfiguration after identifying faulty processors is based on the determination of the coherent cyclic subgraph of the current structure with maximum cardinality [11]. If fault concern a resource processor an another I/O port allocation is required. Then I/O port may be switched to the determined fault-free processor.

A generalized cost of a network traffic with a specified resources deployment and workload of a network is usually tested with experimental methods or simulation. Distributing resources copies in a hypercube with an attempt to optimize system performance measures of interest has been investigated in [5].

The definition $(m, \ d)$ - perfect deployment is a characteristic of the value of the generalized cost of information traffic in the network at a given load of tasks. In the work [14] we have tried to apply the above mentioned approach to a processors network of a 4-dimensional cube type logical structure along with its soft degradation process.

In this paper we have extended the notion of $(m, d|G)$-perfect resources placement in a hypercube type structure G to $(k|G)$-perfect placement. It can be seen as a such allocation of k resources which minimizes the average distances between the working processors and resource processors in the structure G. Also, the algorithm for determining the $(k|G)$-*perfect placement* of resources in the network type hypercube was given.

The main aim of this paper is to apply and compare the above presented approaches to the 4-dimensional cube type processors network along with its soft degradation process. Particularly we were interested in obtaining the values of the average number of working processors for some $(m, d|G)$-resources perfect placement schemes depending on the degree of the network degradation.

The rest of paper consists of two sections and a summary. The second section provides a basic definitions and properties of working structures which are induced by the network in the process of its degradation. There is defined a concept of $(m, d|G)$-*perfect deployment* of processors with resources in the working structure of network G and the $(k|G)$-*perfect placement* . In the third section two algorithms for determining $(m, d|G)$-perfect placement and $(k|G)$-perfect placement are proposed. An illustration of the main algorithms steps for the same structure is given. Finally some concluding remarks are given.

2 The Basic Definitions and Properties

Definition 1. The logical structure of processors network we call the structure of 4-dimensional cube if is described by coherent ordinary graph $G = \langle E, U \rangle$ (E – set of processors, U – set of bidirectional data transmission lines), which nodes can be described (without repetitions) by 4-dimensional binary vectors (labels) in such a way that

$$\left[\delta\big(\varepsilon(e'),\ \varepsilon(e'')\big) = 1\right] \Leftrightarrow [(e',\ e'') \in U] \tag{1}$$

where $\delta\big(\varepsilon(e'),\ \varepsilon(e'')\big)$ is Hamming distance between the labels of nodes e' and e''.

The Hamming distance between two binary vectors $\varepsilon(e')$ and $\varepsilon(e'')$ complies with the dependency:

$$\delta\big(\varepsilon(e'), \varepsilon(e'')\big) = \sum_{k \in \{1,\dots,n\}} (\varepsilon(e')_k \oplus \varepsilon(e'')_k)$$

where:

- $\varepsilon(e')_k$ – the k-th element of the binary vector $\varepsilon(e')$,
- $\oplus$ – modulo 2 sum.

If $|E| = 2^4$ and $|U| = 2|E|$, then such graph we called (non labeled) 4-dimensional cube and denote by H^4.

Denoted by $\widetilde{H}^4 = \langle H^4; \{\varepsilon(e): e \in E(H^4)\}\rangle$ 4-dimensional cube labeled nodes, and by $\breve{G}_p^{\&}(H^4)$ and $\breve{G}_p^{\&}(\widetilde{H}^4)$ – a sets of coherent subgraphs of the graphs H^4 and $\widetilde{H}^4$ of order p, respectively, for class $\& \in \{C, A\}$ (C – cyclic graphs, A – acyclic graphs).

Example 1. The structure G (Fig. 1a) is a structure of type H^4 $\left(G \in \breve{G}_{10}^{A}(H^4)\right)$ because the nodes of this structure can be labeled in accordance with the formula 1 (Fig. 1b).

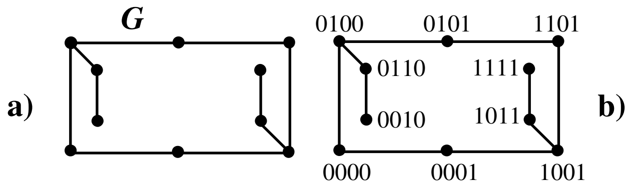

Fig. 1. Illustration of verification that the G structure is a structure of type H^4

The placement of resources in the processors network of 4-dimensional cube type logical structure we will regard as a labeled graph $\langle G; E\rangle$ where $G \in \breve{G}_p(H^4)$ for $p \geq 6$ and $\dot{E} \subset E(G)$ ($\dot{E}$ – set of resource processors, $\{E(G) \setminus \dot{E}\}$ – set of the working processors of the network G).

Denoted by $d(e, e'|G)$ the distance between nodes e and e' in a coherent graph G, that is the length of the shortest chain (in the graph G) connecting node e with the node e'.

Definition 2. We say ([2],[4],[7]) that the labeled graph $\langle G; \dot{E}\rangle$ for $|\dot{E}| \geq 1$ is $(m, d|G)$ - ***perfect placement*** for $m \in \{1, \ldots, \mu(G)\}, d \in \{1, \ldots, D(G)\}, D(G)$ - diameter of the graph G if there exists the set $\dot{E}$ of minimum cardinality such that

$$\left[\forall_{e \in \{E(G) \setminus \dot{E}\}}: \left|\{e' \in \dot{E}: d(e, e'|G) \leq d\}\right| \geq m\right] \wedge \left[\forall_{\{e^*, e^{**}\} \subset \dot{E}}: d(e^*, e^{**}|G) > d\right] \wedge \left[(\mu(e''|G) = 1) \Rightarrow (e'' \notin \dot{E})\right].$$

Denoted by $E^{(d)}(e|G) = \{e' \in E(G): d(e, e'|G) \leq d\}$ for $d \in \{1, \ldots, D(G)\}$ and $\hat{E}^{(1)}(G) = \{e \in E(G): (\exists_{e' \in E(e)}: \mu(e') = 1)\}$.

Denoted by $d_{max}(e, G) = \max_{e' \in E(G)} d((e, e')|G)$, and by $E^{(d)}(e|G) = \{e' \in E(G): d(e, e'|G) = d\}$ for $d \in \{1, \ldots, d_{max}(G)\}$, and by

$$\rho(e, G) = \left(\rho_1(e, G), \ldots, \rho_{d_{max}(e,G)}(e, G)\right) \text{ for } \rho_d(e, G) = \left|E^{(d)}(e|G)\right|. \quad (2)$$

Definition 3. Denoted by $\varphi(e, G) = \sum_{e' \in E(G)} d(e, e'|G)$ for $e \in E(G)$ attainability of the processor e in the network G and by $\Phi(G) = \sum_{e \in E(G)} \varphi(e, G)$ attainability of the network G.

Using (2) we have

$$\varphi(e,G) = \sum_{d=1}^{d_{max(e,G)}} d\rho_d(e,G). \quad (3)$$

Example 2. We determined using (2) - $\rho(e,G)$, and using (3) - $\varphi(e,G)$, and $\Phi(G)$ for structures presented on the Fig. 2. We see that $d_{max(e,G_1)} = d(e_4,e_{10}|G_1) = 7$ and $d_{max(e,G_2)} = 4$. The determined value of $\rho(e,G)$, $\varphi(e,G)$ and $\Phi(G)$ are presented in table 1.

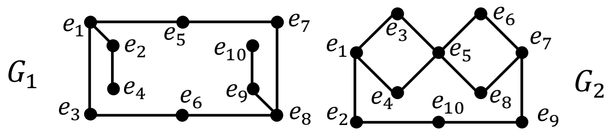

Fig. 2. Example of structures of type H^4 - G_1 $\left(G_1 \in \breve{G}_{10}^{A}(H^4)\right)$ and G_2 $\left(G_2 \in \breve{G}_{10}^{C}(H^4)\right)$

Table 1. The $\rho(e,G)$, $\varphi(e,G)$ and $\Phi(G)$ for the nodes of the structures G_1 and G_2 presented in the Fig. 2

	G_1								G_2				
$d(e,e'\|G)$ / $e \in E(G)$	**1**	**2**	**3**	**4**	**5**	**6**	**7**	$\boldsymbol{\varphi(e,G_1)}$	**1**	**2**	**3**	**4**	$\boldsymbol{\varphi(e,G_2)}$
e_1	3	3	1	1	1	0	0	21	3	2	3	1	20
e_2	2	2	2	1	1	1	0	27	2	3	2	2	22
e_3	2	3	3	1	0	0	0	21	2	4	2	1	20
e_4	1	1	2	2	1	1	1	35	2	4	2	1	20
e_5	2	3	3	1	0	0	0	21	4	2	2	1	18
e_6	2	3	3	1	0	0	0	21	2	4	2	1	20
e_7	2	3	3	1	0	0	0	18	3	2	3	1	20
e_8	3	3	1	1	1	0	0	25	2	4	2	1	20
e_9	2	2	2	1	1	1	0	27	2	3	2	2	22
e_{10}	1	1	2	2	1	1	1	35	2	2	4	1	22
					$\boldsymbol{\Phi(G_1)}$			251			$\boldsymbol{\Phi(G_2)}$		206

Property 1. $\Phi(H^4) = 512$ because $\forall_{e\in E(H^4)}: \left(d_{max}(e,H^4) = 4 \wedge \rho_d(e,H^4) = \binom{4}{d}\right)$. Using (3) we have $\forall_{e\in E(H^4)}: \varphi(e,H^4) = 32$ and $|E(H^4)| = 2^4$, then

$$\Phi(H^4) = |E(H^4)|\,\varphi(e,H^4).$$

Property 2. $d_{max}(e,G) = 8$ when the structure G is a Hamiltonian cycle in H^4.
Property 3. If $G' \subset G''$ that $\{E(G'')\backslash E(G')\} = e^*$, then $\forall_{e\in\{E(G'')\backslash\{e^*\}\}}: \varphi(e,G'') = \varphi(e,G') + d(e,e^*|G'')$.

Definition 4. Denoted by $\psi(e,f) = \sum_{e' \in \dot{E}(f)} d\left(e, e' \middle| G(f)\right)$ for $e \in \dot{E}(f)$ accessibility of the processor e in the placement f and by $\Psi(e,f) = \sum_{e \in \dot{E}(f)} \psi(e,f)$ accessibility of the resource processors in the placement f.

Denoted by $d(f)$ average distances between the working processors and resource processors in the placement $f \in F_k(G)$ for $1 < k < \left\lfloor \left|E\left(G(f)\right)\right|/2 \right\rfloor$. Then

$$d(f) = \left[k\left(\left|E\left(G(f)\right)\right| - k\right)\right]^{-1} \left(\sum_{e \in \{E(G(f))/\dot{E}(f)\}} \sum_{e' \in \dot{E}(G(f))} d\left(e, e' \middle| G(f)\right)\right) \tag{4}$$

where $k = \left|\dot{E}(f)\right|$.

Definition 5. Let $f \in F_k(G)$ be $(k|G)$ – ***perfect placement***, then

$$d(f) = min\{d(f') \colon f' \in F_k(G)\}. \tag{5}$$

Denoted by $\Omega(f)$ distances between the working processors and resource processors in the placement f. Let $\Omega_e^{\Delta}(f)$ denote increment $\Omega(f)$ when add $e \in \left\{E(G) \backslash \dot{E}(f)\right\}$ to the set $\dot{E}(f)$. Then

$$\Omega_e^{\Delta}(f) = \varphi\left(e, G(f)\right) - 2 \sum_{e' \in \dot{E}(f)} d\left(e, e' \middle| G(f)\right). \tag{6}$$

3 The Method and the Algorithm for Determining a Resources Placement

3.1 Determining $(\mathbf{m}, \mathbf{d}|\mathbf{G})$ – Perfect Placement

For G as the first node we choose such a node $\mu(e_i) = \max_{e \in E(G)} \mu(e)$ that the subgraph $\bar{G}^{(d)}(G, e_i) = \left\langle \left\{E(G) \backslash E^{(d)}(e_i)\right\} \right\rangle_G$ has the smallest number of components of coherence. Then we determine a placement for every of these components of coherence, wherein if a component of coherence is one-node it belongs to the set $\dot{E}(f)$.

Based on the presented method was developed the algorithm (I) for determining $(m, d|G)$- ***perfect placement***.

Step 1.

Choose a node $e_i \in E(G)$ such that:

- the degree of $\mu(e_i) = \max_{e \in E(G)} \mu(e)$;
- subgraph $\bar{G}^{(d)}(G, e_i)$ has the smallest number of components of coherence.

Add the node e_i to the set $\dot{E}(f)$.

Step 2.

Check if a component of coherence of the subgraph $\bar{G}^{(d)}(G, e_i)$ is one-node or $\bar{G}^{(d)}(G, e_i) = \emptyset$.

YES

If $\bar{G}^{(d)}(G, e_i) \neq \emptyset$ add all nodes of $\bar{G}^{(d)}(G, e_i)$ to the set $\dot{E}(f)$.

Go to step 3.

NO

Assume that the $\bar{G}^{(d)}(G, e_i)$ is a new graph G.
Return to step 1.

Step 3.

The end of the algorithm (I).

An illustration of the algorithm work is presented in [14]. Algorithm will choose $(1{,}1|G)$ – perfect placement for the structure G_2 from Fig. 2. One of the possible $(1{,}1|G)$-perfect deployment for the structure G (chosen by the algorithm) is shown .

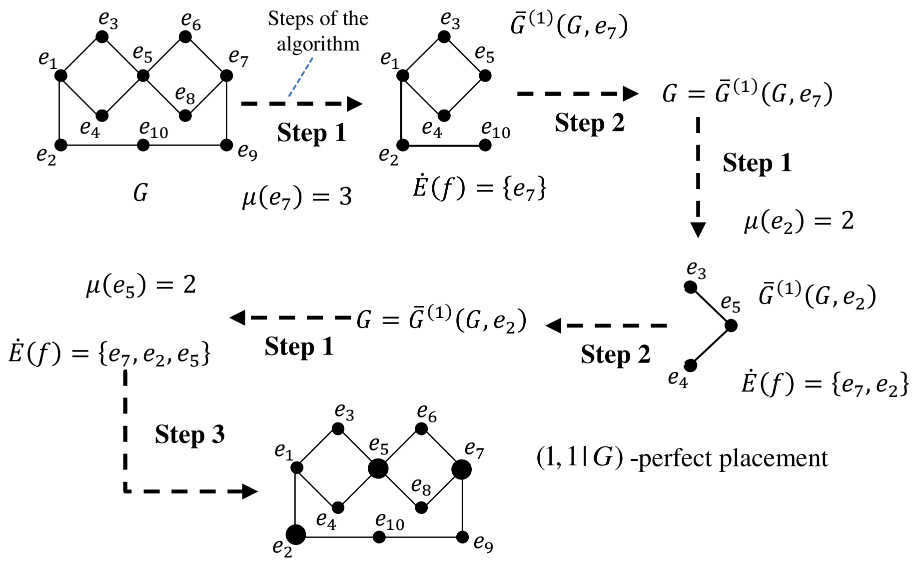

Fig. 3. An illustration of the algorithm (I) steps

3.2 Determining $(k|G)$ – Perfect Placement

The proposed method for determining $(k|G)$ – perfect placement is based on the calculation of $\Omega_e^{\Delta}(f)\colon\{E(G) / \dot{E}(f)\}$ (6). The method choose a node e_i such that $\varphi(e_i, G) = \min_{e \in E(G)} \varphi(e, G)$. The operations of calculation and selection of the node will be repeated for $k < \lfloor |E(G(f))|/2 \rfloor$. For every step is determined $(k+1|G)$ - perfect placement.

Based on the presented method was developed the algorithm (II) for determining $(k|G)$ - ***perfect placement.***

Step 1.

For the structure G using (3) determine $\varphi(e, G)$.

Step 2.

Choose a node $e_i \in E(G)$ such that $\varphi(e_i, G) = \min_{e \in E(G)} \varphi(e, G)$.

Add the node e_i to the set $\dot{E}(f)$.

Step 3.

Check if $k < \lfloor |E(G(f))|/2 \rfloor$.

YES

Using (6) determine $\Omega_e^{\Delta}(f)\colon \{E(G)\,/\,\dot{E}(f)\}$.

Return to step 2.

NO

Assume that the f is $(k|G)$ – ***perfect placement.***

Go to step 4.

Step 4.

The end of the algorithm.

An illustration of the algorithm work is presented in Fig. 4. Algorithm will choose $(k|G)$ – ***perfect placement*** for the structure G_2 from Fig. 2. The algorithm stop working for $k = 4$.

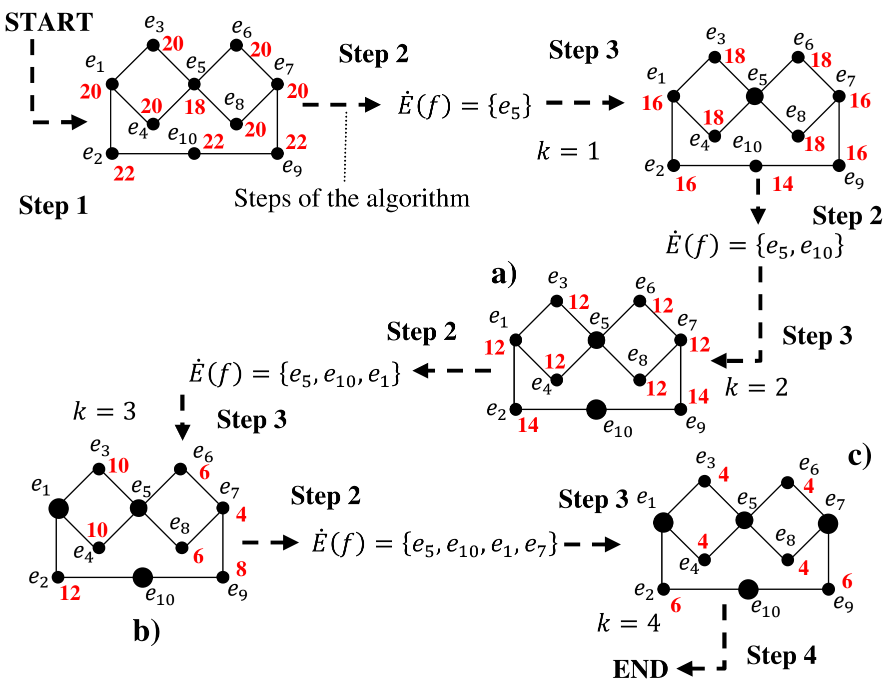

Fig. 4. An illustration of the algorithm (II) steps

The algorithm in 10 steps choose three placements for $k = \{2,3,4\}$ presented on the Fig. 3. a), b), c) respectively.

3.3 Remarks

Comparing the results of operation of both algorithms we can easily observe that the first one gives always $(1, d|G)$– perfect type placements (if such exists) while the second not always allow to obtain placement which is the $(1, d|G)$– perfect type placements. For instance, the placement obtained in the Fig. 4c is $(4|G)$ – perfect placement and simultaneously is $(1,1|G)$ type placement but it is not the perfect placement because of the number of resources is not a set of minimal (is equal to 4 while the algorithm gives us perfect placement with 3 resource processors – Fig. 3).

The algorithm of determining the $(k|G)$ – perfect resources placement may be useful for obtaining approximate solutions for $(1,\delta|G)$– perfect type placements and $\delta\epsilon\{1,..,d_{max}\}$.

We have applied given in 3.2 algorithm (I) for determining the $(1,1|G)$-perfect placements in 62 cyclic working structures (i.e. within the 4-dimensional hypercube network structures with the degradation degree $0 \leq \varrho \leq 7$). By the degradation degree is meant the number equal to $\varrho = 2^4 - |E(G)|$). It was turned out that for some structures the $(1,1|G)$-perfect placements do not exist. Based on the known numbers of instances of these structures [11] in the 4-dimensional hypercube network the average numbers of working processors for the $(1,1|G)$-perfect placement depending on the degree of the network degradation were determined (Table 2) . This value characterizes the loss of the network computing capabilities resulting from the increase of the degree of network degradation.

Table 2. Characterization of the $(1,1|G)$- perfect placement for cyclic working structures in the 4-dimensional hypercube depending on the degree of the network degradation (ϱ)

	ϱ							
	0	**1**	**2**	**3**	**4**	**5**	**6**	**7**
% of existing placements	100	100	100	100	99	100	97	94
average number of working processors	12	11	10	9,19	8,46	8,17	6,92	6

4 Summary

In this paper the method of determining some schemes of resources placement in the 4-dimensional hypercube processors network with soft degradation was presented.

The $(m,d|G)$-perfect placement is a characteristic of a value of the generalized cost of information traffic in the network structure G at a given load of tasks. In the paper this notion was extended to the $(k|G)$-perfect placement which may be easily determined on the base of the accessibility measure of resources calculated as total sum of distances between the working processors and resources processors for the given placement in the working structure G. The algorithm of determining of $(k|G)$ -perfect placement was presented and it may be useful for obtaining approximate solutions for $(1,d|G)$ type perfect placements.

It should be noticed that real cost of information traffic in a network for a given deployment of resources processors depends on the nature of the tasks performed by the network. We plan to examine this problem in the future with the use of simulation methods for a specified (m,d)-perfect deployments and a given type of task load of the network.

References

1. AlBdaiwia, B.F., Bose, B.: On resource placements in 3D tori. Journal of Parallel Distributed Computer 63, 838–845 (2003)
2. AlBdaiwia, B.F., Bose, B.: Quasi-perfect resource placements for two-dimensional toroidal networks. Journal of Parallel Distributed Computer 65, 815–831 (2005)
3. Bae, M.M., Bose, B.: Resource Placement in Torus-Based Networks. IEEE Transactions on Computers 46(10), 1083–1092 (1997)
4. Imani, N., Sarbazi-Azad, H., Zomaya, A.Y.: Resource placement in Cartesian product of networks. Journal of Parallel Distributed Computer 70, 481–495 (2010)
5. Tzeng, N.-F., Feng, G.-L.: Resource Allocation in Cube Network Systems Based on the Covering Radius. IEEE Transactions on Parallel and Distributed Systems 7(4), 328–342 (1996)
6. Chen, H., Tzeng, N.: Efficient resource placement in hypercubes using multiple-adjacency codes. IEEE Trans. Comput. 43(1), 23–33 (1994)
7. Moinzadeh, P., Sarbazi-Azad, H., Yazdani, N.: Resource Placement in Cube-Connected Cycles. In: The International Symposium on Parallel Architectures, Algorithms, and Networks, pp. 83–89. IEEE Computer Society (2008)
8. Chudzikiewicz, J.: Sieci komputerowe o strukturze logicznej typu hiperszęścianu. Instytut Automatyki i Robotyki. Wojskowa Akademia Techniczna, Warsaw (2002) (in Polish)
9. Chudzikiewicz, J., Zieliński, Z.: Determining a non-collision data transfer paths in hypercube processors network. In: Embedded Systems - High Performance Systems, Applications and Projects, pp. 19–34. InTech, Rijeka (2012)
10. Chudzikiewicz, J., Zieliński, Z.: Reconfiguration of a processor cube-type network. Przegląd Elektrotechniczny (Electrical Review) 86(9), 149–153 (2010)
11. Zieliński, Z.: Podstawy diagnostyki systemowej sieci procesorów o łagodnej degradacji i strukturze hiperszęścianu. Wojskowa Akademia Techniczna, Warsaw (2012) (in Polish)
12. Kulesza, R., Zieliński, Z.: The life period of the hypercube processors' network diagnosed with the use of the comparison method. In: Monographs of System Dependability - Technical Approach to Dependability, pp. 65–78. Oficyna Wydawnicza Politechniki Wrocławskiej, Wrocław (2010)
13. Zieliński, Z., Strzelecki, Ł., Kulesza, R.: Diagnosability characterization of the 4 dimensional cube type soft degradable processors' network. In: Monographs on System Dependability – Problems of Dependability and Modeling, pp. 283–296. Oficyna Wydawnicza Politechniki Wrocławskiej, Wrocław (2011)
14. Chudzikiewicz, J., Zieliński, Z.: Resources placement in the 4-dimensional fault-tolerant hypercube processors network. In: Proc. International Conference on Innovative Network Systems and Applications (iNetSApp), Kraków, Poland (2013)
15. Lubkowski, P., Laskowski, D.: The end-to-end rate adaptation application for real-time video monitoring. In: Zamojski, W., Mazurkiewicz, J., Sugier, J., Walkowiak, T., Kacprzyk, J. (eds.) New Results in Dependability & Comput. Syst. AISC, vol. 224, pp. 295–305. Springer, Heidelberg (2013)

16. Reddy, B.A.P.: Design, analysis, and simulation of I/O architectures for hypercube multiprocessors. IEEE Trans. Parallel Distributed Systems 1(2), 140–151 (1990)
17. Kulesza, R., Zieliński, Z.: Metoda generowania struktur logicznych sieci procesorów o łagodnej degradacji typu 4-wymiarowego sześcianu. Biuletyn WAT LX(4) (2011) (in Polish)
18. Byłak, M., Laskowski, D.: Assessment of network coding mechanism for the network protocol stack 802.15.4/6LoWPAN. In: Zamojski, W., Mazurkiewicz, J., Sugier, J., Walkowiak, T., Kacprzyk, J. (eds.) New Results in Dependability & Comput. Syst. AISC, vol. 224, pp. 75–82. Springer, Heidelberg (2013)