Mapping Parallel Programs onto Distributed Computer Systems with Faulty Elements

Mikhail S. Tarkov[1], Youngsong Mun[2], Jaeyoung Choi[2], and Hyung-Il Choi[2]

[1] Fault-tolerant computer systems department, Institute of Semiconductor Physics,
Siberian branch, Russian Academy of Sciences,
13, Lavrentieva avenue, Novosibirsk, 630090, Russia
tarkov@isp.nsc.ru
[2] School of Computing, Soongsil University,
1-1, Sang-do 5 dong, DongJak-gu, Seoul, Korea
{mun, choi, hic}@ computing.soongsil.ac.kr

Abstract. Mapping one-measured ("line", "ring") and two-measured ("mesh", "torus") parallel program structures onto a distributed computer system (DCS) regular structures ("torus", "two-measured circulant", "hypercube") with faulty elements (processor nodes and links) is investigated. It is shown that: 1) one-measured program structures mapped better than two-measured structures; 2) when failures are injected to the DCS structure the one-measured structures mapping aggravated very lesser than the two-measured structures mapping. The smaller a node degree of a program graph and the greater a node degree of a DCS graph the better the mapping quality. Thus, the one-measured program structure's mappings are more fault-tolerant than two-measured structure's one and more preferable for organization of computations in the DCS with faulty elements.

1 Introduction

A Distributed Computer System (DCS) [1-4] is a set of Elementary Machines [EM] that are connected by a network to be program-controlled from these EM. Every EM includes Computing Unit (CU) (a processor with a memory) and a System Device (SD). A functioning of the SD is controlled by the CU and the SD has input and output poles connected by links to output and input poles of v neighbor EM. A structure of the DCS is described by a graph $G_s(V_s, E_s)$, V_s is a set of EM and $E_s \subseteq V_s * V_s$ is a set of links between EM. For the DCS a graph $G_p(V_p, P_p)$ of parallel program is determined usually as a set V_p of the program branches (virtual elementary machines) that communicate with each other by "point-to-point" principle by transmission of messages across logical (virtual) channels (one- or two-directed) from a set $E_p \subseteq V_p * V_p$. In general case numbers (weights) that characterize computing complexities of branches and intensities of communications between neighbor

V.N. Alexandrov et al. (Eds.): ICCS 2001, LNCS 2074, pp. 148–157, 2001.

branches are corresponded to nodes $x, y \in V_p$ and edges (or arcs) $(x, y) \in E_p$ accordingly.

A problem of mapping structure of the parallel program onto structures of DCS is equivalent to the graph isomorphism problem that is NP-complete [5,6]. Now efforts of researchers are directed to search effective heuristics suitable for most cases. In many cases observed in practice of parallel programming the weights of all nodes (and all edges) of program graph can be considered equal to each other and can be neglected .In this case a problem of mapping structure of parallel program onto structure of DCS has the following form [5]. The graph $G_p(V_p, P_p)$ of parallel program is considered as a set V_p and a function

$$G_p : V_p * V_p \to \{0,1\},$$

satisfying

$$G_p(x,y) = G_p(y,x),\ G_p(x,x) = 0$$

for any $x, y \in V_p$. The equation $G_p(x, y) = 1$ means that there is an edge between nodes x and y, that is $(x, y) \in E_p$. Analogously the graph $G_s = (V_s, E_s)$ is determined as a set of nodes (elementary machines (EM)) V_s and a function

$$G_s : V_s * V_s \to \{0,1\}.$$

Here E_s is a set of edges (of links between EM). Suppose that $|V_p| = |V_s| = n$. Let's designate the mapping of parallel program branches to EM by one-to-one function $f_m : V_p \to V_s$.

The mapping quality can be evaluated by the number of program graph edges coinciding with edges of DCS graph. Let's call this number as a cardinality $|f_m|$ of mapping f_m and define it by the expression [5] (the maximum criterion of the mapping quality):

$$|f_m| = (1/2) \sum_{x \in V_p, y \in V_p} G_p(x, y) * G_s(f_m(x), f_m(y)). \qquad (1)$$

The minimum criterion of the mapping quality has the form

$$|f_m| = (1/2) \sum_{x \in V_p, y \in V_p} G_p(x,y) * L_s(f_m(x), f_m(y)). \qquad (2)$$

Here $L_s(f_m(x), f_m(y))$ is equal to the distance between nodes $f_m(x)$ and $f_m(y)$ on the graph G_s.

2 Mapping Algorithm

Let's consider the following approach to the mapping problem [7]. Let initially $f_m(x) = x$. Let $e_p(x)$ be an environment (a set of neighbors) of a node x on the graph G_p and $e_s(x)$ be its environment on the graph G_s. For each node $x \in V_p$ we shall test twin exchanges of all nodes i and j if these nodes are satisfied to the condition

$$i \in e_p(x) \,\&\, i \notin e_s(x) \,\&\, j \in e_s(x) \,\&\, state(j) = 0,$$

where $state(j) = 0$, if a node j has not been exchanged in $e_s(x)$ and $state(j) = 1$ otherwise. Exchanges that don't make worse the performance of mapping f_m will be fixed. This approach is based on assumption about high probability of situation when such exchange increasing the number of nodes $i \in e_p(x)$ in the environment $e_s(x)$ improves (or at least doesn't make worse) the value of performance criterion $|f_m|$. It is obvious that the number of the analyzed exchanges in one evaluation of all nodes $x \in V_p$ doesn't exceed the value $v_p v_s n, n = |V_p|$, where v_p and v_s are maximum degrees of nodes of graphs G_p and G_s accordingly. With $v_p v_s < n$ this approach provides the reduction of computations with respect to the well-known Bokhari algorithm (B-algorithm) [5] and with increase of n the effect of reduction is also increased. On the base of suggested approach the procedure Search has been developed for search of local extremum of function $|f_m|$ in the mapping algorithm (MB-algorithm) that is a modification of the B-algorithm. Besides the MB-algorithm includes following modifications . Firstly initial value of $|f_m|$ is compared with E_p and if the equality is carried out then the algorithm is completed. Secondly check-up of the equality $|f_m| = |E_p|$ executed after every completion of the Search. These examinations also lead very often to a cutting down of the algorithm implementation time.

The MB-algorithm is presented below for the criterion (1). Here: TASK is the description of the graph G_p, VS is the current mapping f_m, BEST is the best found mapping f_m, card(f) is the performance criterion of the mapping f, card *(TASK)* is the performance of the graph G_p selfmapping, BEST $\leftarrow$ VS is the creation of the copy BEST of the mapping VS, Rand-interchange (VS) is the random exchange of n node pairs in the mapping VS, Output(BEST) is the output of the best mapping f_m.

```
Procedure MB;
begin
  done=false; BEST ← VS;
   if (card(VS)=card(TASK)) done=true;
     while (done ≠ true)   do
       begin Search(VS,TASK);
         if (card(VS) > card(BEST))
           begin BEST ← VS; Rand_exchange(VS) end
         else     done=true;
       end
   Output(BEST)
end.
```

The performance of the MB-algorithm has been investigated [7] on the mapping standard parallel program structures ("line", "ring", "mesh") onto regular graphs of DCS (E_2-graph i.e. torus, optimal D_2-graph i.e. two-measured circulant, hypercube) with a number of nodes n varying from 8 to 256. Results of MB-algorithm tests show that the cardinality achieved by the algorithm of mapping graph G_p onto graph G_s with number of nodes $n=|V_p|=|V_s| \leq 256$ is no less than 90% of the number of nodes in G_p for cases when G_p is the "line" or the "ring" and no less than 80% for the case when the G_p is the “mesh”. The MB-algorithm reduces the time of mapping with respect to B-algorithm in two exponents for mapping one-measured structures ("line", "ring") onto E_2- and D_2-graphs and in exponent for mapping two-measured structures ("mesh") onto hypercube. For all that the MB-algorithm doesn't make worse the performance of the mapping with respect to the B-algorithm.

In this paper we investigate the MB-algorithm quality for mapping parallel program structures onto structures of distributed computer systems (DCS) with faulty elements. The mapping quality is investigated for different DCS network topologies (structure types) and for different numbers of faulty processors and links (interconnections).

Usually the DCS graphs initially are regular (torus, hypercube etc) but during the DCS functioning the failures in processors and links can be arised. As a result the parallel program must be remapped onto the nonregular connected functioning part of the DCS. The decentralized operating system of the DCS provides the equality of program branches count to the DCS functioning processor count, i.e. $|Vs|=|Vp|$ and the computational load is evenly distributed among functioning processors [3,4]. In other words, the parallel program can be adjusted to a number of processors in the DCS.

3 Conditions for Mapping Parallel Programs onto DCS with Faulty Processors and Links

How many failures can be tolerant in a DCS for mapping parallel program? There are two types of bounds for the tolerant multiplicity *t* of failures.

First, a number of faulty nodes must be no more than *(n-1)/2, n=|V|*. This bound is determined by selfdiagnosability condition [2]. In accordance with the bound, selfdiagnostics algorithm works correctly only when the nonfaulty part of the system has no less than half of the nodes number.

Second, the graph with faulty nodes and edges must be connected. This property is necessary for communications between branches of the parallel program. The probability p_t of the DCS graph disconnection and the probability $p_c = 1 - p_t$ of the disconnection absence were investigated with respect to the number of faulty nodes and edges. Investigation was realized by Monte Carlo simulation of regular graphs with faulty nodes and edges and the simulation results are presented below.

Figs.1-6 show the simulation plots of p_t versus the percentage of failed nodes (Figs.1,3,5) and edges (Figs.2,4,6) for a torus (Figs.1,2), a circulant (Figs.3,4) and hypercube (binary cube) (Figs.5,6) with n=64 nodes.

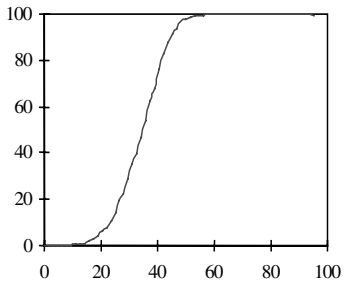

Fig.1

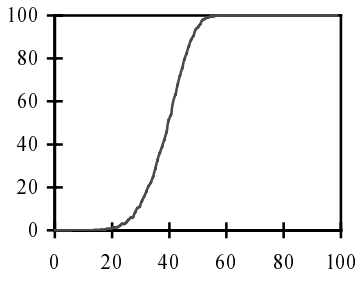

Fig.2

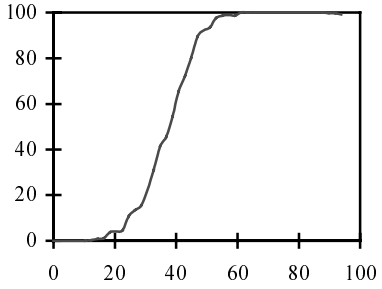

Fig. 3

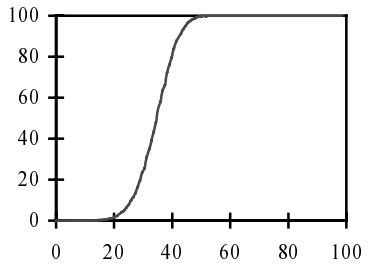

Fig. 4

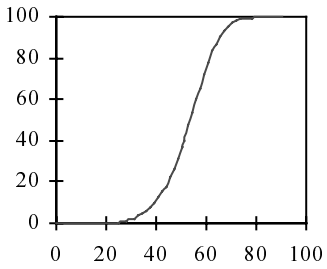

Fig. 5

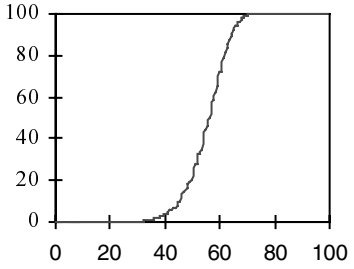

Fig. 6

Simulation results show that:

1)for tori and two-measured circulants $p_t \approx 1$ for *t>n/2,* and for hypercubes $p_t \approx 1$ for *t>3n/4*;

2) for tori and two-measured circulants $p_c \approx 1$ for $t \le 0.2n$ (see Figs.1-4) , and for hypercubes $p_c \approx 1$ for $t \le 0.3n$ (see Figs.5,6);

These results are true both for graphs with faulty nodes and for graphs with faulty edges. In the second case *n* is a number of the DCS edges. They show that graph topologies with higher node degree allow the network to sustain a larger number of failures before disconnection. Thus for $t \le 0.2n < (n-1)/2$ all considered topologies are practically connected and we can apply the above mapping algorithm.

4 Investigation of the Mapping Algorithm

Four types of parallel programming graphs G_p were considered: “line”, “ring”, “mesh” and “torus”. These graphs with the number of nodes from *n*=8 to 256 were mapped onto the DCS structure (torus, two-measured circulant (D2-graph), hypercube) with faulty nodes or faulty edges.

The plan of the mapping algorithm investigation is:

1.Create initial graph G_s with a given regular topology and a given number of nodes.

2. Define a multiplicity of failures t and generate a set of k distorted DCS graphs $G_s(i)$, $i=1,...,k$ by means of injection of failures to nodes or to edges of the initial graph.

3. Map the program graph G_p onto every distorted DCS graph $G_s(i)$ and calculate the mapping quality $| f_m |_i$ for every mapping .

4. Calculate average value of relative mapping quality

$$r_m = \frac{1}{kE_p} \sum_{i=1}^{k} |f_m|$$

Results of the mapping algorithm simulation can be distinguished on two groupes:
1. mapping “line” and “ring”;
2. mapping “torus” and “mesh”.

Table 1. Minimal values $r_m(V) / r_m(E)$ of average mapping qualities for faulty nodes and for faulty edges

G_p / G_s	Torus	Circulant	Hypercube
Line	0.802 / 0.803	0.801 / 0.802	0.868 / 0.867
Ring	0.793 / 0.796	0.797 / 0.790	0.859 / 0.850
Mesh	0.571 / 0.792	0.538 / 0.790	0.633 / 0.613
Torus	0.500 / 0.595	0.495 / 0.468	0.586 / 0.586

For the first group (Table.1), if the number of faulty nodes $t < 0.2n$ than the average mapping quality is no less than 0.8 when the DCS graph has topology of torus or circulant and is no less than 0.86 when the DCS graph has topology of hypercube.

For the second group, if the number of faulty nodes $t < 0.2n$ than the average mapping quality is no less than 0.46 when the DCS graph has topology of torus or circulant and is no less than 0.58 when the DCS graph has topology of hypercube.

Figs.7-10 show the simulation plots of the r_m versus the number n of the DCS nodes for several values of the percentage of failed nodes (Tables. 2-5).

Table 2. Mapping quality, "line → torus" mapping

	16	36	64	100	144	169	225
0%	1	0.971	0.905	0.909	0.923	0.929	0.938
10%	0.937	0.864	0.828	0.842	0.836	0.843	0.856
20%	0.926	0.839	0.804	0.802	0.813	0.810	0.813

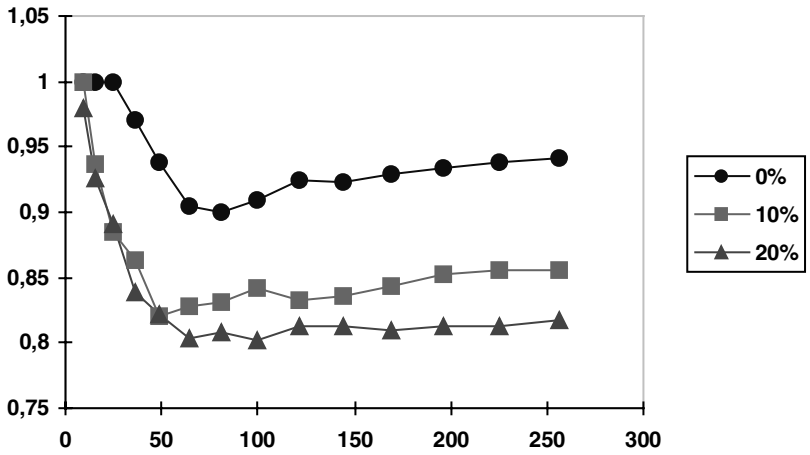

Fig. 7. Mapping quality, "line → torus" mapping

Table 3. Mapping quality, "mesh → torus" mapping

	16	36	64	100	144	196	256
10%	0.865	0.698	0.674	0.639	0.633	0.641	0.628
20%	0.783	0.652	0.626	0.577	0.575	0.566	0.558

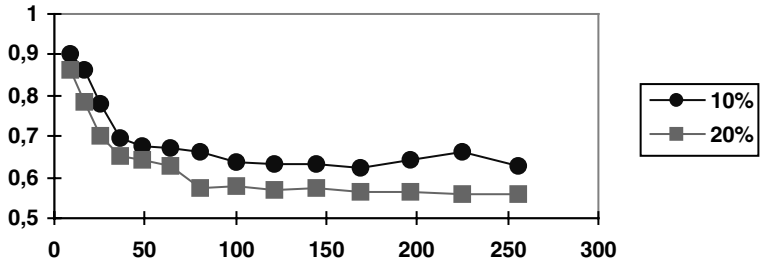

Fig. 8. Mapping quality, "mesh → torus" mapping (r_m=1 for 0% percentage of failed nodes)

Table 4. Mapping quality, "line → hypercube" mapping

	8	16	32	64	128	256
0%	1	1	1	0.968	0.937	0.937
10%	1	0.948	0.902	0.887	0.883	0.895
20%	0.919	0.909	0.880	0.868	0.870	0.878

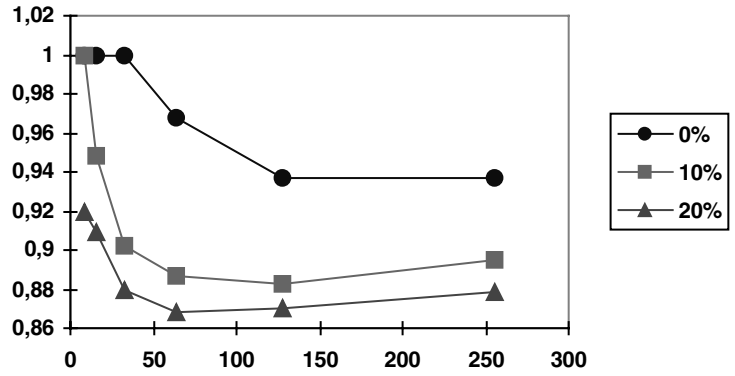

Fig. 9. Mapping quality, "line → hypercube" mapping

Table 5. Mapping quality, "mesh → hypercube" mapping

	8	16	32	64	128	256
0%	1	1	0.923	0.812	0.776	0.771
10%	0.875	0.857	0.794	0.743	0.699	0.665
20%	0.875	0.805	0.742	0.719	0.661	0.633

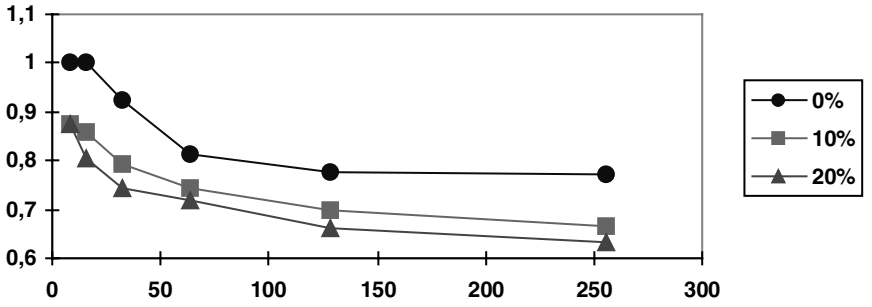

Fig. 10. Mapping quality, "mesh → hypercube" mapping

5 Conclusion

Mapping typical graphs (“line”,”ring”,”mesh”,”torus”) of parallel programs onto regular graphs (“torus”, “two-measured circulant”, “hypercube”) of distributed computer system (DCS) is considered when the DCS has failures such as faulty processors (nodes) or faulty links (edges). It is shown by the mapping algorithm simulation that:

1) one-measured program structures (line and ring) are mapped better than two-measured structures (mesh and torus);

2) the one-measured structures mapping is changed to worse very less (6 – 15% for 20% faulty elements) than the two-measured structures mapping(15-45%).

For hypercube the mapping quality values are greater than for torus and two-measured circulant . The smaller node degree of a program graph and the greater node degree of a DCS graph the greater the mapping quality. Thus, the one-measured program structure’s mappings are more fault-tolerant than two-measured structure’s one and more preferable for organization of computations in the DCS with faulty elements.

Acknowledgement

This work was supported by the BK21 program (E-0075).

References

1. Korneev, V.V.: Architecture of Computer Systems with Programmable Struc ture.Novosibirsk, Nauka (1985)(in Russian).
2. Dimitriev, Yu.K.: Selfdiagnostics of modular computer systems. Novosibirsk, Nauka (1994) (in Russian)
3. Korneev, V.V., Tarkov, M.S.: Operating System of Microcomputer System with Programmable Structure MICROS, Microprocessornie sredstva i sistemy (Microprocessor means and systems). 4 (1988) 41-44 (in Russian)
4. Tarkov, M.S.:Parallel Fault Tolerant Image Processing in Transputer System MICROS-T. Nauchnoe priborostroenie. Vol.5. 3-4 (1995) 74-80 (in Russian)
5. Bokhari, S.H.: On the Mapping Problem, IEEE Trans. Comput., C-30(3), (1981) 207-214
6. Lee, S.-Y., Aggarval, J.K.:A Mapping Strategy for Parallel Processing, IEEE Trans. Comput., C-36(4), (1987) 433-442
7. Tarkov, M.S.: Mapping Parallel Programs Onto Distributed Robust Computer Systems. Proceed. of the 15th IMACS World Congress on Scientific Computation, Modelling and Applied Mathematics, Berlin, August 1997. V.6. Application in Modelling and Simulation. Proc. Ed. By Achim Sydow. (1997) 365-370