



A New Heuristic Approach for Reliability Optimization of Distributed Computing Systems Subject to Capacity Constraints*

RUEY-SHUN CHEN,** DENG-JYI CHEN AND Y. S. YEH

Institute of Computer Science and Information Engineering

National Chiao Tung University, Hsinchu, Taiwan, R.O.C.

(Received November 1993; accepted December 1993)

Abstract—Distributed Computing Systems (DCS) have become a major trend in computer system design today, because of their high speed and reliable performance. Reliability is an important performance parameter in DCS design. In the reliability analysis of a DCS, the term of K -Node Reliability (KNR) is defined as the probability that all nodes in K (a subset of all processing elements) are connected.

In this paper, we propose a simple, easily programmed heuristic method for obtaining the optimal design of a DCS in terms of maximizing reliability subject to a capacity constraint. The first part of this paper presents a heuristic algorithm which selects an optimal set of K -nodes that maximizes the KNR in a DCS subject to the capacity constraint. The second part of the paper describes a new approach that uses a K -tree disjoint reduction method to speed up the KNR evaluation. Compared with existing algorithms on various DCS topologies, the proposed algorithm finds a suboptimal design much more efficiently in terms of both execution time and space than an exact and exhaustive method for a large DCS.

Keywords—Distributed computer system (DCS), K -node reliability (KNR), Capacity constraint, Heuristic, K -tree disjoint reduction.

1. INTRODUCTION

Distributed Computing Systems (DCS) have become increasingly popular in recent years, for the advent of VLSI technology and low-cost microprocessors has made distributed computing economically practical in today's computing environment. The DCS provides potential increases in reliability, throughput, fault tolerance, resource sharing, and extendibility [1,2]. Achieving increases in these performance characteristics, however, requires careful design to increase the reliability of a DCS. A DCS has been defined as a collection of nodes at which reside computing resources that communicate with each other via a set of links [3]. Large scale DCSs are coming into use primarily because of the economy achieved through resource sharing [4]. The main objective of a DCS is to provide efficient communication among various nodes in order to increase their utility and to make their service available to more users [5]. One of the fundamental considerations in designing such systems is that of system reliability, which strongly depends on the topological layout of the communication links [6]. Reliability is a very good measure of DCS performance if all the needed network users are to be connected with each other. Several DCS reliability

*This research was partially supported by the National Science Council of the Republic of China under contract NSC82-0301-E009-013.

**Author to whom all correspondence should be addressed.

measures have been defined and associated evaluation methods have been developed. One of these distributed system reliability measure [6–8], K -Node Reliability (KNR), is adopted in this paper, KNR is defined to be the probability that all K (a subset of the processing elements) nodes in the DCS can be run successfully. A DCS may be modelled by a graph in which the nodes correspond to the file servers and the edges to the communication links.

Several heuristic methods [9–11] have been proposed for obtaining an optimal network topology that gives the maximum overall reliability of a given computer communication network. All of the proposed methods find an approximate solution, because as the number of links increases, the number of possible layouts of the links grows faster than exponentially. To date, the problem of maximizing the reliability of a DCS under a capacity constraint have been considered by [12].

In this paper, we present a heuristic algorithm for maximizing reliability by the node select problem to obtain an optimal design DCS. In the second part of the paper, we present a new method for computing KNR based on a K -tree disjoint reduction method. The method is simple and easy to program. This algorithm may be useful when the system under consideration is very large and when any near optimal solution serves the purpose.

The organization of the rest of this paper is as follows. In Section 2, the problem statement, notation, and definitions that will be used throughout this paper are given. Section 3 presents the mathematical formulation of the problem. The heuristic algorithm with the K -tree disjoint reduction method are proposed in Section 4. In Section 5, simulation and analyses of the heuristic algorithm with the K -tree disjoint reduction method is discussed. Section 6 concludes the paper.

2. PROBLEM STATEMENT, NOTATION AND DEFINITIONS

PROBLEM STATEMENT. The method suggested here can be characterized as follows:

GIVEN.

Network topology.

The reliability of each communication link.

The capacity of each node installed.

Each node is perfectly reliable.

Each link is either in the working (ON) state or failed (OFF) state.

CONSTRAINT. The total capacity constraint on the DCS.

GOAL. Maximize reliability for optimal design of a DCS.

NOTATION AND DEFINITIONS. The notation and definitions used in the rest of the paper are summarized here.

$G(N, L)$	an undirected DCS graph in which the set of nodes N represents the PEs and the links L represent the communication links	X_s	the starting node of the DCS
N_i	a node representing a processing element i	C_i	the capacity of the i^{th} node
L_i	an edge representing a communication link i	C_s	the capacity constraint in a DCS
X_i	a decision node, $X_i = 1$ if i is selected, else $X_i = 0$	G_x	denotes the graph G with X -node specified
X^*	denote the first vector following X in the numerical ordering that has the property that $X \leq X^*$	$R(G_x)$	the reliability of X -node solution of the DCS graph G
S_i	the i^{th} node select to add to the starting node set	G_k	denotes the graph G with K -nodes specified, and $K \geq 2$
$X = \{X_1, X_2, X_3, \dots, X_n\}$	the set of decision nodes, $X_i = 0$ or $1, i = 1, 2, \dots, n$	$R(G_k)$	the reliability of K -nodes solution of the DCS graph G
		e_i	the i^{th} edge of DCS
		X^\wedge	denotes the vector which has current optimal solution $R(G_x)$
		R^\wedge	denotes the current optimal reliability solution

DEFINITION 1. A **K -tree** is a tree of G covering all nodes of K such that its pendant nodes are in K .

DEFINITION 2. **K -Node Reliability (KNR)** is defined as the probability of successful communication, i.e., all K -nodes (a subset of all the processing elements, $K \geq 2$) in a DCS are connected by working edges.

DEFINITION 3. A **K -node DCS reliability problem** is the problem of computing $R(G_k)$. The problem is a member of the class of number K -complete problems, which is a class of NP-hard problems not known to be in NP.

3. MATHEMATICAL FORMULATION OF THE PROBLEM

The problem considered in this paper may be stated as follows: Determine an optimal DCS that gives maximum reliability with the given capacity constraint. In other words, we are to find a set of K -nodes from the given set of N nodes which constitutes an optimal DCS in that KNR is maximized, and the total capacity satisfies the needed total capacity constraint. Consider the simple optimal DCS problem stated mathematically as follows:

$$\begin{aligned} & \text{Maximize } R(G_k) \\ & \text{subject to: } \sum_{X_i \in K} C_i X_i \geq C_s. \end{aligned}$$

To find an optimal solution, we do not consider an exhaustive method, since it is too time-consuming, instead, the exact algorithm to find an optimal solution by finding maximum reliability K -nodes and executing only a portion of all the combinations. The exact algorithm is given below.

STEP 1. Determine whether the count number has overflowed. If $X > \text{overflow}_X$ then stop; otherwise go to the next step.

STEP 2. Compare the capacity $\sum_{i=1}^n C_i (X^* - 1)_i$ with capacity constraint C_s . If $\sum_{i=1}^n C_i (X^* - 1)_i < C_s$, then substitute X for X^* and go to back to Step 1, else go to the next step.

STEP 3. Compare the capacity $\sum_{i=1}^n C_i X_i$ with capacity constraint C_s . If $\sum_{i=1}^n C_i X_i < C_s$, then $X = X + 1$ and go to back to Step 1, else go to the next step.

STEP 4. Compare the vector partial ordering X with X^\wedge . If $X \geq X^\wedge$, then X is substituted for X^* and go to back to Step 1, else go to the next step.

STEP 5. Compute reliability $R(G_x)$ and compare with current optimal solution R^\wedge . If $R(G_x) > R^\wedge$, then substitute R^\wedge for $R(G_x)$ and X^\wedge for X , else substitute X for X^* and go to back to Step 1.

STEP 6. Continue loop until X overflow. Then the last $R(G_{x^\wedge})$ reliability computed with the SYREL [13] is obtained for our K -node reliability and X^\wedge is obtained for our optimal node vector.

The exact method is illustrated below by a numerical example. Consider the six node DCS with eight links depicted in Figure 1. Here our problem is to determine an optimal DCS which includes some of the nodes X_1, X_2, \dots, X_6 , whose total capacity exceeds the capacity constraint of 70 units. The optimization can be formulated as the following mathematical problem:

$$\begin{aligned} & \text{Maximize } R(G_k) \\ & \text{subject to: } \sum_{X_i \in K} C_i X_i \geq 70. \end{aligned}$$

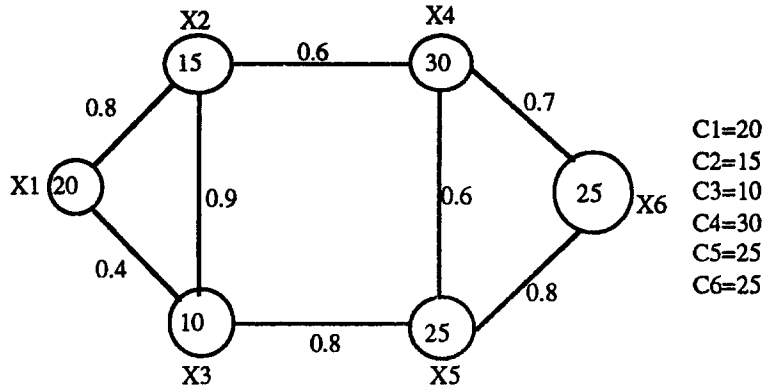


Figure 1. A six-node distributed computing system.

Using a C computer program based on the proposed algorithm, an optimal DCS was obtained by an Intel-486 personal computer in 3 clock cycles of execution time. The optimal K -DCS topology with node vector (X_1, X_2, X_3, X_5) was found in the DCS with maximum reliability of 0.7628 and total capacity of 70 units. The evaluation count was only 17, compared with a count of 64 for the exhaustive method. This is a rather modest reduction in computing, but a much greater execution time will be spent in problems with a larger DCS. Although an exact method [12] provides an optimal solution, it cannot effectively reduce the problem space. Sometimes an application requires a fast way to compute reliability because of its resource considerations. In this situation, obtaining the optimal reliability may not be the best idea, instead, a fast method providing near optimal reliability computation may be better. In real cases, most DCSs are large, and as the number of nodes increases, the execution time needed to obtain a solution grows exponentially. So to reduce the total execution time for optimal design of a DCS, we were motivated to develop a heuristic algorithm using the K -tree disjoint reduction method to reduce the execution time needed to compute KNR when the DCS under consideration is very large.

4. THE PROPOSED HEURISTIC ALGORITHM AND K -TREE DISJOINT REDUCTION METHOD

Obviously, the problem for a large DCS, such as a metropolitan area network, requires a great deal of execution time. Our problem is to propose a good heuristic using the K -tree disjoint reduction method so that the DCS can provide the desired performance. Because of computational advantages, a heuristic method may be preferred to an exact method when the DCS is large. A number of greedy algorithms for computer communication networks have already been proposed by Aggarwal *et al.* [9–11]. The main drawback of their approach is that it requires the generation of spanning trees, many of which are never used subsequently, because of the cost constraint. A good deal of computer time is thus wasted.

In this section, we suggest a heuristic algorithm for large DCS problems that largely avoids unnecessary generation of spanning trees. We regard the DCS as a weighted graph, in which the weight of each node represents its capacity, and generate K -node disjoint terms using a K -tree disjoint reduction method to obtain KNR. The proposed heuristic algorithm is as follows.

Heuristic Algorithm

The algorithm consists of six steps:

STEP 0. Initialization, read system parameters.

STEP 1. Choose any one node as a starting node X_s and compute the reliability of the starting node to other nodes $R(G_k)$ using SYREL [13].

STEP 2. Take the sum of the node capacity of both the starting node and the other nodes $\sum_{X_i \in K} C_i X_i$.

STEP 3. Select the node X_i with the maximum $[R(G_k) * \sum_{X_i \in K} C_i X_i]$ and add to the next starting node set.

STEP 4. Use the K -tree disjoint reduction method to reduce immediately disjoint terms and call SYREL [13] to compute the KNR of the next starting node set.

STEP 5. Repeat Step 1 to Step 4 until the total capacity of the starting node set exceeds the capacity constraint. Then output the last starting node set and the maximum reliability.

The operation of the heuristic algorithm is illustrated in Figure 2. We determine the optimal design of a DCS that maximizes KNR, iff the total capacity of the corresponding starting node set satisfies the given capacity constraint.

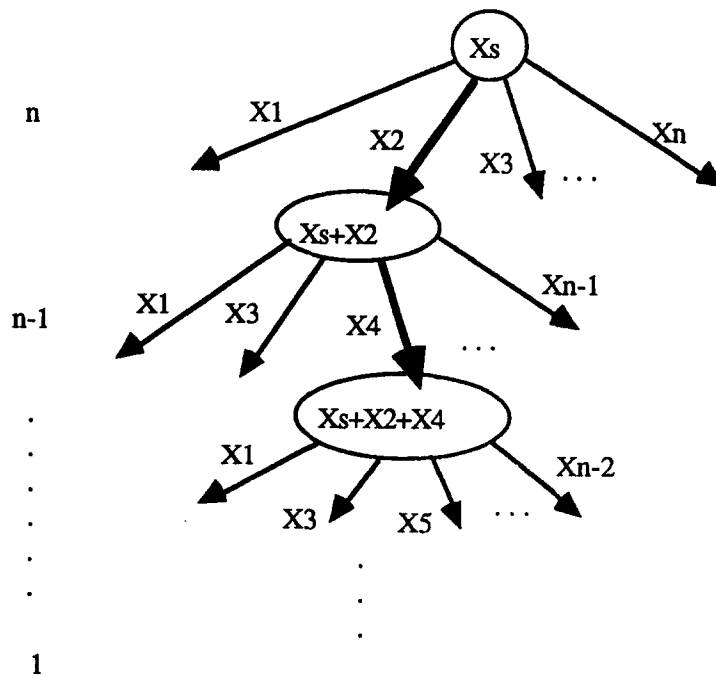


Figure 2. The operation of the heuristic algorithm.

The K-Tree Disjoint Reduction Method

Several reliability measures have been studied by researchers in the context of DCSs, including source-to-multiple-terminal reliability (SMT) [3], the survivability index [14], computer network reliability [15], and multiterminal reliability [16]. SMT and computer network reliability are popular and useful measures for DCSs. The problem with the survivability index is that the computation time for this index is prohibitively large, even for relatively small networks [14]. The multiterminal reliability algorithm works well as long as there are not too many alternate paths between source and destination pairs. However, these reliability analyses are not directly applicable to the reliability analysis of DCSs without appropriate modification.

In this paper, to speed up the proposed heuristic algorithm, we present a new reliability evaluation method, which we call the K -tree disjoint reduction method, that employ different concept to efficiently compute the KNR. It is based on the K -tree disjoint reduction method, which reduces immediately disjoint terms and, hence, reduces the computation time.

Consider the simple DCS in Figure 3. There are four processing elements (X_1, X_2, X_3, X_4) connected by links e_1, e_2, e_3, e_4 , and e_5 . Let nodes X_1 and X_2 be the starting node set in

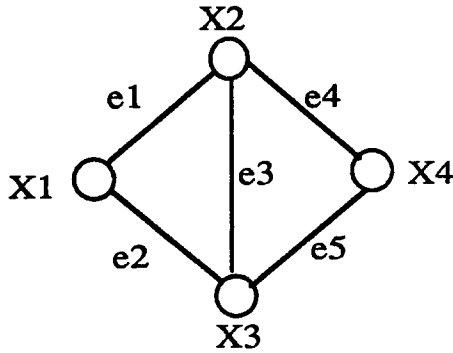


Figure 3. A simple distributed computing system.

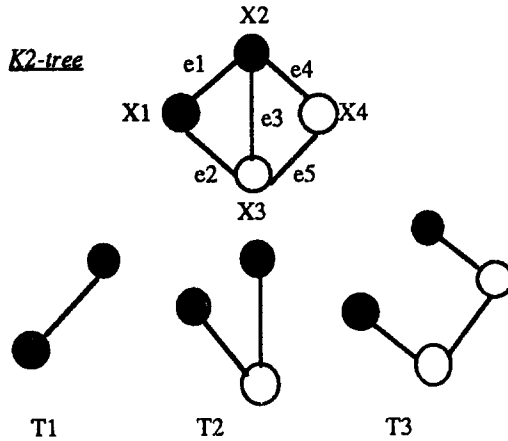


Figure 4. The K_2 -tree with $K_2 = \{X_1, X_2\}$ in simple DCS.

the DCS. We can identify some K_2 -trees with $K_2 = \{X_1, X_2\}$ rooted on X_1 from the DCS in Figure 4 as follows:

- (1) $X_1 \ e1 \ X_2 \ \dots \dots \dots T_1$
- (2) $X_1 \ e2 \ X_3 \ e3 \ X_2 \ \dots \dots \dots T_2$
- (3) $X_1 \ e2 \ X_3 \ e5 \ X_4 \ e4 \ X_2 \ \dots \dots \dots T_3$

$$K_2 = \{X_1, X_2\}$$

$$R(G_{k_2}) = \Pr(T_1 \cup T_2 \cup T_3)$$

$$\vdots$$

$$= \text{disjoint terms } D_{k_2}.$$

We get three K_2 -trees $T_1, T_2,$ and T_3 with $K_2 = \{X_1, X_2\}$ in Figure 4, where $R(G_{k_2})$ is the probability of the K_2 -tree and D_{K_2} is the total disjoint terms of the K_2 -tree. Consider Step 3 of the proposed heuristic algorithm. In Figure 5, we add node X_3 to the starting-nodes set. The disjoint terms of the K_3 -tree are $T_4, T_5, T_6, T_7,$ and T_8 . Therefore, the probability of the K_3 -tree is $R(G_{k_2} \cup \{X_3\}) = \Pr(T_4 \cup T_5 \cup T_6 \cup T_7 \cup T_8)$, equal to $\Pr\{D_{k_2} \cap [\cup \text{path}(X_3 \rightarrow K_2)]\}$. The disjoint terms of $\cup \text{path}(X_3 \rightarrow K_2) = e2 \cup e3 \cup e4e5$ are much less than the expansion of $(T_4 \cup T_5 \cup T_6 \cup T_7 \cup T_8)$, since $D_{k_2} \cap [\cup \text{path}(X_3 \rightarrow K_2)]$ will generate less and shorter disjoint terms, and $(T_4 \cup T_5 \cup T_6 \cup T_7 \cup T_8)$ generate on exponential increase in disjoint terms as the size of the K -trees increases. So the proposed algorithm can save much execution time when the DCS is large.

The expanded terms of $(T_4 \cup T_5 \cup T_6 \cup T_7 \cup T_8)$ may contain some nodes in common, since they are not mutually exclusive in general. When the nodes are completely reliable, $D_{k_2} \cap [\cup \text{path}(X_3 \rightarrow K_2)]$ can generate fewer and shorter disjoint terms to evaluate KNR directly, because the

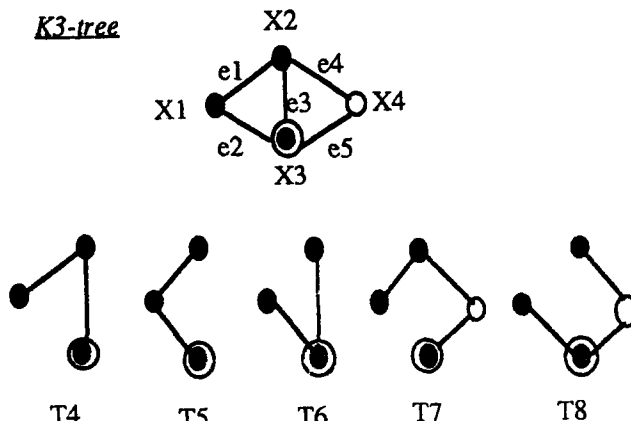


Figure 5. The K_3 -trees with $K_3 = \{X_1, X_2, X_3\}$ in simple DCS.

terms generated in the expansion of $D_{k2} \cap [\cup \text{path } (X_3 \rightarrow K_2)]$ use the K -tree disjoint reduction method and the AND operator.

$$\begin{aligned}
 R(G_{k2} \cup \{X_3\}) &= \Pr(T_4 \cup T_5 \cup T_6 \cup T_7 \cup T_8) \\
 &\vdots \\
 &\Rightarrow \dots = \Pr\{D_{k2} \cap [\cup \text{path } (X_3 \rightarrow K_2)]\},
 \end{aligned}$$

where $\cup \text{path } (X_3 \rightarrow K_2) = e_2 \cup e_3 \cup e_4 e_5$.

Now we define several rules for finding $\cup \text{Path } (X_i \rightarrow K_j)$ as follows:

DEFINITION.

- P_i^0 = all the paths from X_i to $\{X_s\}$
- P_i^1 = all the paths from X_i to $\{X_s, S_1\}$
- P_i^2 = all the paths from X_i to $\{X_s, S_1, S_2\}$
- P_i^{j+1} = $\{P_i^j \mid \text{not include } X_{(j+1)} \text{ path}\} \cup \{P_i^j \mid \text{include } S_{(j+1)} \text{ path, but delete string behind } S_{(j+1)}\}$

Numeric Example of K -Tree Disjoint Reduction Method

We use the DCS shown in Figure 6 as an example to show how the $\cup \text{path } (X_3 \rightarrow K_2)$ works.

STEP 1. Select node X_1 as the starting-node, then generate from the other nodes all path sets P_2^0, P_3^0, P_4^0 to the starting node as follows:

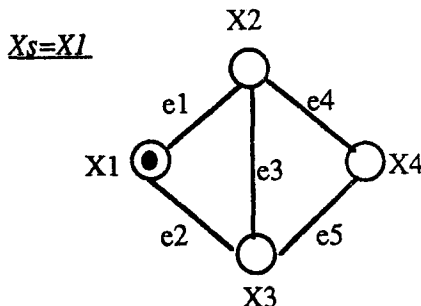


Figure 6. A simple DCS with $X_s = X_1$.

P_2^0 : /* All the paths from X_2 to X_1 */

- (1) $X_2 e_1 X_1$
- (2) $X_2 e_3 X_3 e_2 X_1$
- (3) $X_2 e_4 X_4 e_5 X_3 e_2 X_1$

P_3^0 : /* All the paths from X_3 to X_1 */

- (1) $X_3 e_2 X_1$
- (2) $X_3 e_3 X_2 e_1 X_1$
- (3) $X_3 e_5 X_4 e_4 X_2 e_1 X_1$

P_4^0 : /* All the paths from X_4 to X_1 */

- (1) $X_4 e_4 X_2 e_1 X_1$
- (2) $X_4 e_5 X_3 e_2 X_1$
- (3) $X_4 e_4 X_2 e_3 X_3 e_2 X_1$
- (4) $X_4 e_5 X_3 e_3 X_2 e_1 X_1$

STEP 2. Select node X_2 to add to the starting-node set $\{X_1, X_2\}$ as show in Figure 7.

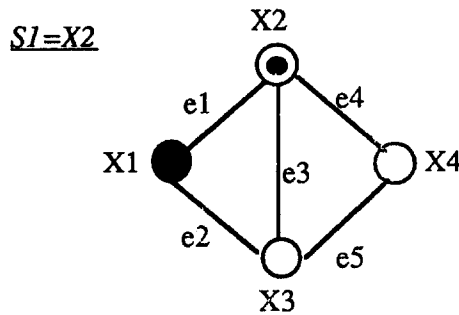


Figure 7. A simple DCS with the starting-node set $\{X_1, X_2\}$.

P_3^1 : /* All the paths from X_3 to $\{X_1, X_2\}$ are equal to \cup path ($X_3 \rightarrow K_2$) is easily generate from P_3^0 */

- (1) $X_3 e_2 X_1$
- (2) $X_3 e_3 X_2$
- (3) $X_3 e_5 X_4 e_4 X_2$

P_4^1 : /* All the paths from X_4 to $\{X_1, X_2\}$ are equal to \cup path ($X_4 \rightarrow K_2$) is easily generated from P_4^0 */

- (1) $X_4 e_4 X_2$
- (2) $X_4 e_5 X_3 e_2 X_1$
- (3) $X_4 e_5 X_3 e_3 X_2$

STEP 3. Select node X_3 to add to the starting-node set $\{X_1, X_2, X_3\}$ as show in Figure 8.

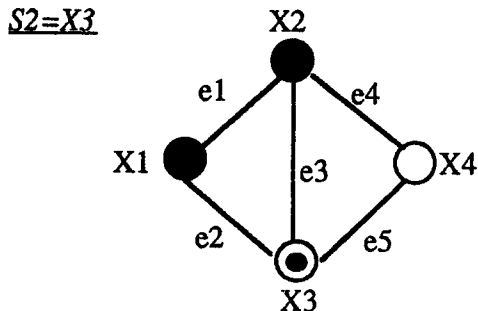


Figure 8. A simple DCS with the starting node set $\{X_1, X_2, X_3\}$.

P_4^2 : /* All the paths from X_4 to $\{X_1, X_2, X_3\}$ are equal to \cup path ($X_4 \rightarrow K_3$) is easily generate from P_4^1 */

- (1) $X_4 e_5 X_3$
- (2) $X_4 e_4 X_2$

EXAMPLE. Consider again the six-node DCS with eight links depicted in Figure 1. Using a C computer program based on the proposed heuristic algorithm, we obtain the result shown in

Figure 9. The selecting node depending on maximum $[R(G_k) * \sum_{X_i \in K} C_i X_i]$. The algorithm takes 1 clock cycle of execution time on an Intel-486 personal computer. Obviously, although it gets a suboptimal solution, the algorithm saves much execution time compares with the exact method [12]. The K -DCS topology with node vector (X_1, X_4, X_5) was found in the DCS with maximum reliability of 0.7532 and total capacity of 75 under a capacity constraint of 70 units.

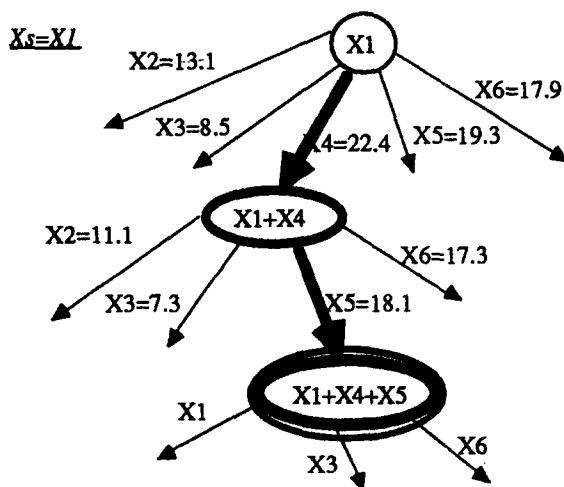


Figure 9. The result of the heuristic algorithm for the DCS in Figure 1.

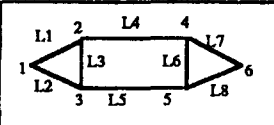
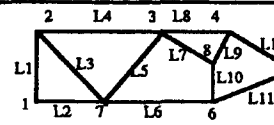
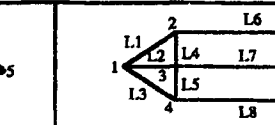
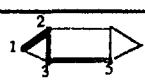
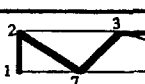
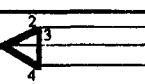
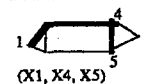
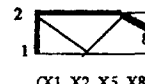
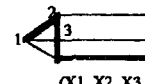
5. SIMULATION AND ANALYSIS

In order to evaluate the performance of our heuristic approach, we applied the proposed algorithm to a wide variety of DCS problems. The execution time results for the heuristic algorithm were compared with those for the exact method [12]. Reliabilities of the K -node DCS were evaluated by applying the K -tree disjoint reduction method and SYREL [13]. To verify the efficiency of our algorithm, simulation programs were implemented in C programming language on an Intel-486 PC/AT. A portion of the simulation results are depicted in Table 1.

From the simulation results, it is evident from the table that the nature of the optimal design of a DCS depends on the DCS topology. Unlike computer network reliability problems, which are static-oriented, the KNR problems in the DCS are dynamic-oriented, since many factors (node capacity, DCS topology) can greatly affect the efficiency of the algorithm [17]. Thus it is very difficult to quantify the time complexity exactly. The proposed heuristic algorithm employs a new approach to the K -tree disjoint reduction method, which effectively speeds up the whole reliability evaluation. A comparison can be made based on the immediately disjoint term. From such a comparison, one can tell approximately how much space and execution time units are required for a particular DCS. The heuristic method saves more execution time than the exact method for a large DCS, and its complexity of only $O(2^e * N^2)$ is better than the $O(2^e * 2^n)$ of the exact method, where e is the number of edges and n is the number of nodes.

It seems that the proposed algorithm is straightforward and reasonable. The K -tree disjoint reduction method generates fewer disjoint terms than traditional reliability index as such as source to multiterminal reliability [3] and multiterminal reliability [16]. For example, assume that P_i^0 is generated, it implies that $P_i^1, P_i^2, \dots, P_i^{j+1}$ all can be easily generated from P_i^0 to keep the whole system operational. But in the traditional reliability index, the expanded terms of the K -trees may contain some nodes in common, since they are not mutually exclusive in general. When the nodes are completely reliable, $D_{k2} \cap [\cup \text{path}(X_3 \rightarrow K_2)]$ can generate less and shorter disjoint terms to evaluate KNR directly, because the terms generated in the expansion of $D_{k2} \cap [\cup \text{path}(X_3 \rightarrow K_2)]$ use the K -tree disjoint reduction method and the AND operator. Thus, the heuristic algorithm takes less computer time than the optimal exact and exhaustive method on various DCS topologies which are listed in Table 1. From the results, we find that our

Table 1. Optimal designs for various DCS topologies.

DCS Topology							
DCS Given data (X1 as start-node)	Node #	1 2 3 4 5 6	1 2 3 4 5 6 7 8	1 2 3 4 5 6 7 8			
	Capacity	20 15 10 30 25 25	20 30 18 26 40 20 10 35	55 60 70 35 45 55 60 40			
	Link #	1 2 3 4 5 6 7 8	1 2 3 4 5 6 7 8 9 10 11 12	1 2 3 4 5 6 7 8 9 10 11 12 13			
	Reliability	.8 .4 .9 .6 .8 .6 .7 .8	.9 .4 .8 .4 .9 .5 .7 .9 .8 .9 .8 .9 .9	.9 .8 .9 .8 .9 .8 .9 .8 .9 .8 .9 .8 .9			
Capacity constraint	$C_s \geq 70$		$C_s \geq 100$		$C_s \geq 200$		
Exhaustive method	Evaluation count	32		128		128	
Exact method (optimal)	Optimal topology	 (X1, X2, X3, X5)	 (X1, X2, X3, X4, X7)	 (X1, X2, X3, X4)			
	Maximum reliability	0.7628		0.8245		0.9854	
	Execution time	3 clock *		106 clock *		2454 clock *	
	Total capacity	70 unit		104 unit		220 unit	
	Evaluation count	17		78		75	
Heuristic method	Optimal topology	 (X1, X4, X5)	 (X1, X2, X5, X8)	 (X1, X2, X3, X7)			
	Maximum reliability	0.7532		0.7994		0.9827	
	Execution time	1 clock *		29 clock *		490 clock *	
	Total capacity	75 unit		125 unit		245 unit	

algorithm performs more efficiently than the exact and exhaustive method in all cases and can easily be transformed into a computer program. Thus, it is an optimal design for large DCSs.

6. CONCLUSION

Distributed computing systems have become very popular for their high fault-tolerance, potential for parallel processing, and reliable performance. One of the important issues in the design of a DCS is reliability. In this paper, we have presented a heuristic algorithm that uses the *K*-tree disjoint reduction method to determine the optimal *K*-nodes in a DCS so as to maximize the KNR under a capacity constraint. Traditional reliability indexes such as source-to-multiterminal reliability [3], survivability [14], multiterminal reliability [16], and so on, are not directly applicable for the analysis of distributed reliability in a DCS. Thus, a new approach for the optimal design *K*-nodes of the DCS must be developed.

The proposed heuristic algorithm is based on a systematic node selection approach that determines the optimal design of a DCS by maximizing the KNR under a capacity constraint. Because we apply straightforward heuristic rules, our algorithm generates fewer disjoint terms than traditional reliability indexes such as source to multiterminal reliability and requires less execution time and space than the exact and exhaustive method [12]. We conclude that our algorithm is an efficient tool for finding an optimal design for large DCSs.

REFERENCES

1. V.K. Prasanna Kumar, S. Hariri and C.S. Raghavendra, Distributed program reliability analysis, *IEEE Trans. Software Eng.* **12** (1), 42–50 (1986).
2. A. Kumar, S. Rai and D.P. Agrawal, Reliability evaluation algorithms for distributed systems, In *Proc. IEEE INFOCOM 88*, pp. 851–860, (1988).
3. A. Satyanarayana and J.N. Hagstrom, New algorithm for reliability analysis of multiterminal networks, *IEEE Trans. Reliability* **30**, 325–333 (1981).
4. D.J. Chen and T.H. Huang, Reliability analysis of distributed systems based on a fast reliability algorithm, *IEEE Trans. on Parallel and Distributed Systems* **3** (2) (1992).
5. L. Fratta and U.G. Montanari, Synthesis of available networks, *IEEE Trans. Reliab.* **R-25**, 81–87 (1976).
6. R.S. Wilkov, Design of computer networks based on a reliability measure, In *Proceedings of the Symposium on Computer Communication. Networks and Teletraffics*, pp. 371–384, (1986).
7. E.L. Lawler and M.D. Bell, A method for solving discrete optimization problems, *Ops. Res.* **14**, 1098–1111 (1966).
8. J.A. Stankovic, A perspective on distributed computer systems, *IEEE Trans. Comput.* **33**, 1102–1115 (1984).
9. K.K. Aggarwal, Y.C. Chopra and J.S. Bajwa, Topological layout of links for optimizing the S-T reliability in a computer communication system, *Microelectron. Reliab.* **22**, 341–345 (1982).
10. K.K. Aggarwal, Y.C. Chopra and J.S. Bajwa, Topological layout or links for optimizing the overall reliability in a computer communication system, *Microelectron. Reliab.* **22**, 347–351 (1982).
11. Y.C. Chopra, B.S. Sohi and K.K. Aggarwal, Network topological for maximizing the terminal l reliability in a computer communication system, *Microelectron. Reliab.* **24** (5), 911–913 (1984).
12. R.-S. Chen, D.J. Chen and Y.S. Yeh, Reliability optimization of distributed computing systems subject to capacity constraint, *Computers Math. Applic.* (to appear).
13. S. Hariri and C.S. Raghavendra, SYREL: A symbolic reliability algorithm based on path and cutset methods, USC Tech. Rep., (1984).
14. R.E. Merwin and M. Mirherkerk, Derivation and use of survivability criterion for DDP systems, In *Proc. 1980 Nat'l. Computer Conference*, pp. 139–146, (1980).
15. K.K. Aggrawal and S. Rai, Reliability evaluation in computer-communication networks, *IEEE Trans. Reliability* **30**, 32–35 (1981).
16. A. Grnarov and M. Gerla, Multiterminal reliability analysis of distributed processing system, In *Proc. 1981 Int'l. Conf. on Parallel Processing*, pp. 79–86, (1986).
17. D.-J. Chen and M.S. Lin, New reliability evaluation algorithms for distributed computing system, *Journal of Computer* (1992) (to appear).