J. Parallel Distrib. Comput. 75 (2015) 123–132

Contents lists available at ScienceDirect

J. Parallel Distrib. Comput.

journal homepage: www.elsevier.com/locate/jpdc

Constructing all shortest node-disjoint paths in torus networks

Cheng-Nan Lai

Department of Information Management, National Kaohsiung Marine University, Kaohsiung, 81157, Taiwan

HIGHLIGHTS

- An optimal construction of all shortest node-disjoint paths in a torus network.
- High probability of the existence of disjoint shortest paths in a torus was shown.
- Finding disjoint shortest paths in a torus is equivalent to that in a hypercube.

ARTICLE INFO

Article history:
Received 4 March 2014
Received in revised form
31 August 2014
Accepted 6 September 2014
Available online 16 September 2014

Keywords:
Torus
k-ary n-cube
Mesh
Hypercube
Node-disjoint paths
Optimization problem

ABSTRACT

An n-dimensional torus network, also called wrap-around mesh or toroidal network, is a Cartesian product of n cycle networks. In particular, it was named k-ary n-cube when the sizes of the n cycle networks are all equal to k. In this paper, m node-disjoint shortest paths from one source node to other m (not necessarily distinct) destination nodes are constructed in an n-dimensional torus network, provided the existence of such node-disjoint shortest paths which can be verified in $O(mn^{1.5})$ time, where m is not greater than the connectivity. The worst-case time and space complexities of the construction procedure are both optimal $O(mn)$. In the situation that all of the source node and destination nodes are mutually distinct, brute-force computations show that the probability of the existence of the m node-disjoint shortest paths (from the source node to the m destination nodes) in a k-ary n-cube is greater than 94%, 70%, 91%, 69%, and 89% for $(k, n, m) = (2, 7, 7), (3, 4, 8), (4, 3, 6), (5, 3, 6)$, and $(6, 3, 6)$, respectively.

1. Introduction

Due to advanced hardware technology, it is now feasible to build a large-scale multiprocessor system consisting of hundreds or even thousands of processors. One crucial step in designing a multiprocessor system is to determine the topology of its interconnection network (network for short) where nodes and links correspond to processors and communication channels, respectively. Since the network topology plays a significant role in system performance, many possible options have been proposed in the literature. One of the most practical network topologies is the torus network (torus for short) [13,14,26] because many multiprocessor systems have been built based on it [1,18,23].

An n-dimensional torus (abbreviated to an n-torus) is a Cartesian product of n cycle networks (cycles for short) of sizes greater than one. The number of nodes of an n-torus is the product of the sizes of the n cycles, and each node has a label of an n-digit number with each digit's value ranging from 0 to the size of its corresponding cycle minus one. Two nodes are connected by a link if and only if their labels differ by exactly one digit and the two different digits' values are next to each other (i.e., ± 1 mod the size of the digit's corresponding cycle). Fig. 1 shows the structure of a 2-torus which is a Cartesian product of two cycles of sizes 5 and 6. Both the degree and connectivity of an n-torus are $2n$ minus the number of cycles of size 2. The *connectivity* of a network is the minimum number of nodes whose removal can make the network disconnected or trivial [5]. Note that a cycle of size 2 can be recognized as a network consisting of two nodes connected by a link.

An n-torus has been further renamed to the k-ary n-cube [2,3,6,9,30] in case that the sizes of the n cycles are all equal to k, where $k \geq 2$. In particular, when $k = 2$, a 2-ary n-cube is just identical to an n-dimensional hypercube (abbreviated to an n-cube) [20,19,27]. Fig. 2 shows the structure of a 4-cube where two disjoint 3-cubes, i.e., a left 3-cube (consisting of eight nodes labeled with trailing 0) and a right 3-cube (consisting of eight nodes labeled with trailing 1), are connected by eight cross links. In addition, each node has a label of four bits (or binary digits) such that each bit's position corresponds to a dimension.

Routing is a process of transmitting messages among nodes/processors. Its efficiency and reliability, which are crucial

E-mail address: cnlai@mail.nkmu.edu.tw.

http://dx.doi.org/10.1016/j.jpdc.2014.09.004
0743-7315/

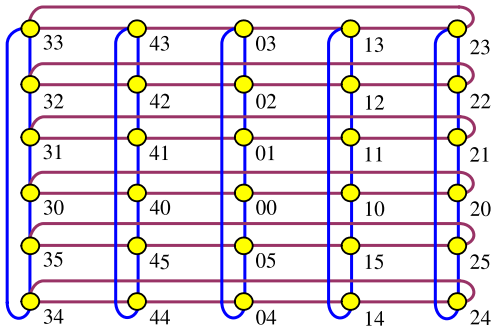

Fig. 1. The structure of a 2-torus which is a Cartesian product of two cycles of sizes 5 and 6.

to the system performance, can be enhanced by employing internally node-disjoint paths (disjoint paths for short), because they can be used to avoid congestion, accelerate transmission rate, and provide alternative transmission routes. Two paths are *internally node-disjoint* if they do not share any common node except their end nodes. In order to reduce the transmission latency and cost, routing with disjoint paths [7,8,14,15,24,25] expects their maximal length and total length to be minimized, respectively, where the *length* of a path is the number of links in it. The benefits of disjoint paths make them play an important role in the study of routing, reliability, and fault tolerance in parallel and distributed systems [10–13,17,19–22,28,29].

There are three categories of disjoint paths, i.e., one-to-one, one-to-many, and many-to-many [11]. Suppose that W is a network with connectivity n. According to Menger's theorem [5], there exist n disjoint paths between every two distinct nodes of W. They belong to the one-to-one category. Many one-to-one disjoint paths constructed for a variety of networks can be found in the literature [10,13,14,22]. On the other hand, according to Theorem 2.6 in [4], there exist n disjoint paths from one node to other n distinct nodes in W. They belong to the one-to-many category. One-to-many disjoint paths were first studied in [25] where the Information Dispersal Algorithm (IDA for short) was proposed on the hypercube. By taking advantages of disjoint paths, the IDA has numerous potential applications to secure and fault-tolerant storage and transmission of information. Some examples of one-to-many disjoint paths can be found in [7,8,12–15,17,21,28,29]. Many-to-many disjoint paths (or named set-to-set disjoint paths), which connect two sets of nodes in W, can be found in [15,16].

Routing functions have been shown effective in deriving disjoint paths in hypercube-like networks [19–21]. By the aid of routing functions, it was shown in [19] that m disjoint shortest paths from one source node to other m (not necessarily distinct) destination nodes can be constructed in an n-cube, provided the existence of such disjoint paths which can be verified in $O(mn^{1.5})$ time (also see [8,24]), where $m \leq n$. The time and space complexities of the construction procedure are both optimal $O(mn)$. In this paper, we study the problem of constructing disjoint shortest paths from one source node to many other destination nodes in an n-torus. Based on our idea, this problem was first transformed into a corresponding problem of constructing disjoint shortest paths with special properties in a $2n$-cube, and then its solutions are applied to derive the required paths. By the aid of the construction procedure of [19], it will be shown that m disjoint shortest paths from one source node to other m (not necessarily distinct) destination nodes can be constructed in an n-torus, provided the existence of such disjoint shortest paths which can be verified in $O(mn^{1.5})$ time, where m is not greater than the connectivity of the n-torus. In addition, the time and space complexities of our construction procedure (of this paper) are both optimal $O(mn)$.

In order to figure out the probability that there exist disjoint shortest paths in an n-torus, brute-force computations were carried out for verifying all the combinations of the source node and destination nodes by running computer programs. In the situation that all of the source node and destination nodes are mutually distinct, computational results show that the probability of the existence of the m disjoint shortest paths (from the source node to the m destination nodes) in a k-ary n-cube is greater than 94%, 70%, 91%, 69%, and 89% for $(k, n, m) = (2, 7, 7)$, $(3, 4, 8)$, $(4, 3, 6)$, $(5, 3, 6)$, and $(6, 3, 6)$, respectively. Since IDA [25] relies heavily on one-to-many disjoint paths, the construction of disjoint shortest paths in an n-torus is not only theoretically interesting but also practical in real applications, especially when all shortest one-to-many disjoint paths come into use.

The rest of this paper is organized as follows: In the next section, routing functions are revisited and the construction of all shortest disjoint paths in an n-cube is described. As shown in Section 3, a necessary and sufficient condition for the existence of disjoint shortest paths in an n- torus is described in terms of routing functions, and the construction of them is also provided. Section 4 shows the detailed computational results. In Section 5, this paper concludes with some remarks on the efficiency of the construction procedure. For the brevity of this paper, whenever we discuss time/space complexity, we mean worst-case time/space complexity.

2. An optimal construction of disjoint shortest paths in an n-cube

Suppose that s is the source node and $d_1, d_2, \ldots, d_m$ are m (not necessarily distinct) destination nodes in an n-cube, where $m \leq n$

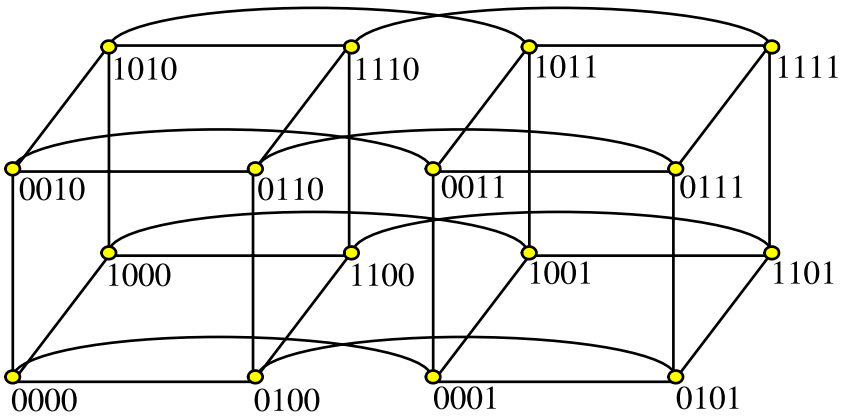

Fig. 2. The structure of a 4-cube.