The Journal of Supercomputing (2022) 78:17758–17775
https://doi.org/10.1007/s11227-022-04557-6

Vertex transitivity, distance metric, and hierarchical structure of the dual-cube

Pranava K. Jha[1]

Accepted: 20 April 2022 / Published online: 23 May 2022

Abstract

The dual-cube, derivable from the hypercube, admits a number of good properties that render it as a good network topology, especially when the node size exceeds several million. This paper presents several other welcome characteristics of the graph. Prominent among them are: (1) vertex transitivity that facilitates the working of an algorithm meant for a "local" context in the global context as well, (2) an exact formula for the distance between two nodes, which leads to a precise result on the distance-wise node distribution of the graph and an exact formula for the average node distance, and (3) a hypercube-like hierarchical structure of the graph that is amenable to an inductive treatment.

Keywords Dual-cube · Interconnection networks · Network topology · Vertex transitivity · Shortest distance and hierarchical structure

1 Introduction

The *dual-cube* is a special kind of a more general *network topology*, called the *metacube* (introduced by Li et al. [14]) that itself is derivable from the *hypercube*. The idea has been to mitigate the problem of the rapid increase in the degree of the hypercube when the node size exceeds several million.

With the same node degree n, the dual-cube contains 2^{n+1} as many nodes as the hypercube Q_n, and with the same number of nodes, it has about 50% fewer edges [22]. Remarkably, it retains most good characteristics of the latter, notably, high connectivity, high fault tolerance, low diameter, and easy routing [14].

✉ Pranava K. Jha
pkjha384@hotmail.com

[1] St. Cloud, MN 56304, USA

1.1 Objectives of the paper

The central objective of this paper is to present certain key results that significantly enhance the importance of the dual-cube from both theoretical point of view and the engineering point of view. Here are the salient points.

1. The first major objective is to prove that that the graph is vertex-transitive.

   A vertex-transitive graph offers a huge advantage, viz. "local" algorithms on it would work globally as well, since all vertices hold equivalent roles in the global context. Furthermore, vertex-transitive graphs are more strongly connected than other regular graphs. Not surprisingly, most network topologies in use today (notably the hypercube, the torus, and the circulants) are vertex-transitive and, to a lesser extent, edge-transitive. (Definitions of the technical terms appear in Sect. 1.2.)
2. The second objective is to present an algorithm that traces a shortest path between two nodes in the dual-cube.

   The distance metric plays an important role in the efficiency of routing algorithms and leads to the average node distance that is an important quality measure of a network.
3. The third objective is to present a scheme that generates a sequence of graphs $\langle G_0, \dots, G_{2m} \rangle$, where G_0 is the two-node graph $0 - 1$ and G_{i+1} is obtainable by introducing a matching between two copies of G_i, and where G_{2m} is isomorphic to DQ_m. (The situation is somewhat similar to the recursive structure of the hypercube.)

1.2 Definitions and preliminaries

A graph connotes a finite, simple, undirected and connected graph. Let G be a graph, and let dist(u, v) denote the (shortest) distance between vertices u and v in G [21]. Further, let dia(G) denote the *diameter* of G [21].

For n-bit strings x and y, let H(x, y) denote the *Hamming distance* between the two. The n-dimensional hypercube Q_n (also called the n-cube) is the graph on the vertex set $\{0, 1\}^n$, where nodes x and y are adjacent iff $\mathrm{H}(x, y) = 1$.

Let $x \cdot y$ (or xy) denote the concatenation of the binary strings x and y, and let $\overline{a} := 1 - a$, where $a \in \{0, 1\}$. Next, let $x \veebar y$ denote the n-bit string obtainable by the bitwise XOR operation between two n-bit strings x and y. It is easy to see that $\veebar$ is both commutative and associative.

Definition 1.1 For an n-bit string $x = b_{n-1} \dots b_0$ (so $0 \le x \le 2^n - 1$ in decimal), let $x^{(i)}$ be the n-bit string obtainable from x by replacing b_i by $\overline{b_i}$, where $0 \le i \le n - 1$.

Definition 1.2 For $m \ge 1$, the dual-cube DQ_m is a spanning subgraph of the hypercube Q_{2m+1}. Its edge set is given by $E_0 \cup E_1 \cup E_2$, where

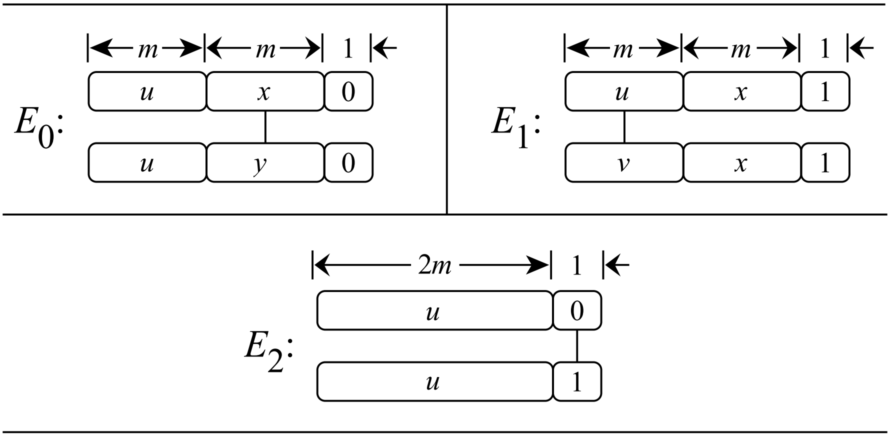

Fig. 1 The three edge types of DQ_m, vide Definition 1.2

1. $E_0 = \{\{uv0, uv^{(0)}0\}, \ldots, \{uv0, uv^{(m-1)}0\} \mid |u| = |v| = m\}$
2. $E_1 = \{\{uv1, u^{(0)}v1\}, \ldots, \{uv1, u^{(m-1)}v1\} \mid |u| = |v| = m\}$, and
3. $E_2 = \{\{x0, x1\} \mid |x| = 2m\}$.

Definition 1.3 The nodes of DQ_m are distinguishable into two types, as follows:

- *Type 0*: those that are of the form $x0$ (binary) or $2i$ (decimal), and
- *Type 1*: those that are of the form $x1$ (binary) or $2i + 1$ (decimal).

Let $e \in E(\mathrm{DQ}_m)$. Call e an edge of Type i if $e \in E_i$, $0 \leq i \leq 1$, and call e a *cross edge* if $e \in E_2$. See Fig. 1 for a depiction of the three edge types. Meanwhile, a node of the hypercube/dual-cube is viewable both as a binary string, say, x and as the corresponding nonnegative integer $N(x)$, a formula for which appears in Equation (1).

$$N(x) = \begin{cases} 0 & \text{if } x = 0 \\ 1 & \text{if } x = 1 \\ 2^{|v|}N(u) + N(v) & \text{if } x = uv, \text{ where } |u|,\ |v| \geq 1. \end{cases} \tag{1}$$

It is easy to see that DQ_m is a regular graph of degree $m + 1$. Figure 2 presents four different drawings of DQ_2. Among other things, they show that the graph admits (1) an embedding on the torus without any edge crossing, (2) an edge decomposition into a Hamiltonian cycle and a perfect matching, (3) a vertex partition into four copies of DQ_1, i.e., C_8, and a vertex partition into eight copies of the four-cycle.

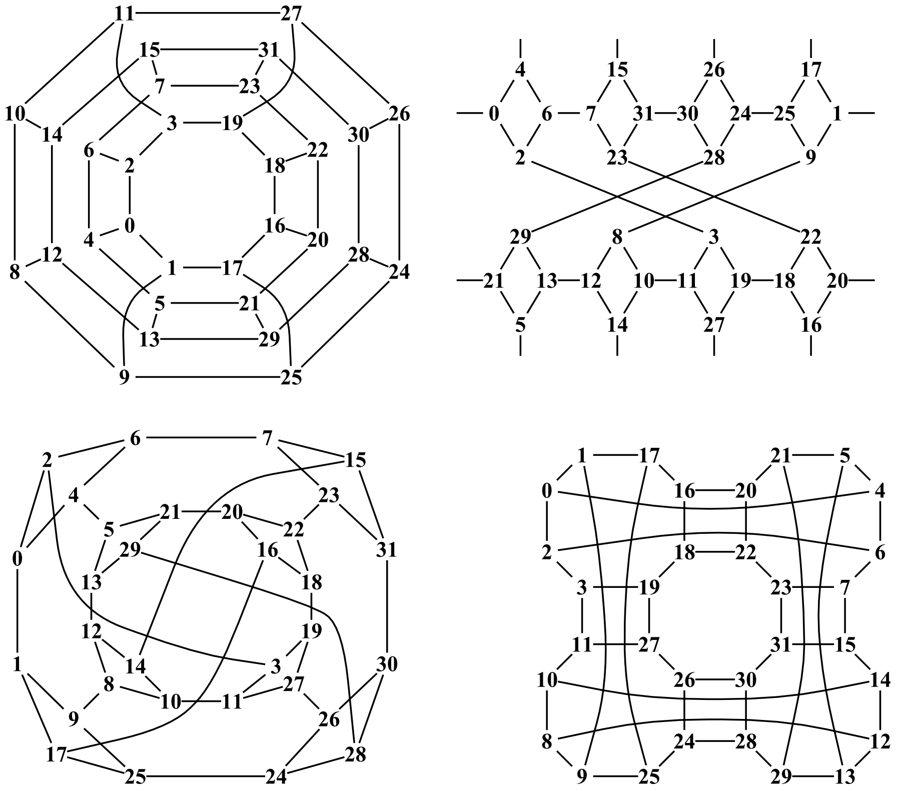

Fig. 2 Four drawings of DQ_2

Definition 1.4

1. A graph is said to be vertex-transitive if for every pair of vertices u and v, it admits an automorphism that sends u to v.
2. A graph is said to be edge-transitive if for every pair of edges $\{u, v\}$ and $\{x, y\}$, it admits an automorphism f such that $f(\{u, v\}) = \{x, y\}$.

The concepts of vertex transitivity and edge transitivity are distinct. In particular, a vertex-transitive graph is necessarily regular, whereas an edge-transitive graph need not be regular. If a graph is both vertex-transitive and edge-transitive (called a *symmetric graph*), then its vertex connectivity as well as its edge connectivity is optimally equal to its degree.

For a set S of integers and an integer j, let $S + j$ denote the set $\{i + j \mid i \in S\}$, and let $C(n, i)$ denote the binomial coefficient $\binom{n}{i}$, where $n \geq i \geq 0$. For any undefined term, see West [21].

1.3 Hypercubes and related networks

The hypercube possesses an extremely rich structure [6, 20]. For example, it is symmetric, distance-regular, and Hamiltonian decomposable. Further, its diameter is logarithmic relative to its order, and it is amenable to emulating a wide variety of other frequently used networks. Not surprisingly, it has been one of the most popular interconnection networks. For example, the architectures of Intel iPSC, the nCUBE, the Connection Machine CM-2, and SGI Origin 2000 are based on the hypercube [14].

Despite its attractive features, the hypercube suffers from certain practical problems. In particular, its scalability is an issue. (A scalable network is one whose size may be increased in such a way that changes to the existing configuration are minor, and performance degradation, if at all, is negligible.) In response, size-scalable network topologies such as mesh/torus, ring, and the tree have been used in practice, despite their inherently limited topological characteristics such as low connectivity, large diameters, large average node distances, and lack of fault tolerance.

At the other end of the spectrum, when the number of nodes in a network exceeds several million (which is a reality in the ever-expanding digital world today), increase in the degree of the hypercube becomes an issue. To mitigate this problem, Loh et al. [16] presented the exchanged hypercube. See [3, 5, 7, 9, 10, 25] for certain related studies. Working along this line of research, Li et al. [14] introduced a regular network (also derivable from the hypercube), called the metacube, of which the dual-cube is a special case. Interestingly, the dual-cube is a special kind of the exchanged hypercube as well.

A number of other hypercube-related networks are discussed by Li et al. [14] and Pai and Chang [18].

1.4 Studies specific to the dual-cube

Here are some of the prominent results that appeared on various aspects of the dual-cube during the past two decades.

1. Jha [8] presented a 1-perfect code in DQ_m, where $m = 2^k - 2$.
2. Lai and Tsai [11] showed that every node of DQ_m lies on a cycle of every possible even length and that the graph maintains this property even if it has up to $m - 1$ edge faults. (The result is optimal with respect to the number of edge faults.)
3. Li et al. [12] established that collective communications may be carried out in the dual-cube almost as efficiently as in the hypercube.
4. Shih et al. [19] proved that DQ_m admits $m + 1$ mutually independent Hamiltonian cycles. (The result is optimal since each vertex of this graph has exactly $m + 1$ neighbors.)
5. Wu and Wu [22] showed that DQ_m admits $\lfloor (m-1)/2 \rfloor$ edge-disjoint Hamiltonian cycles.

6. Chen and Tsai [2], Li et al. [13], and Zhang et al. [24] presented certain results on fault tolerance and fault diagnosability of the dual-cube.

Interestingly, the dual-cube itself admits certain variants. See Angjeli et al. [1] and Chen and Kao [4].

1.5 Comparison of interconnection networks

Comparing interconnection networks is tricky, since there is no single network that is the best for all applications and all operating environments, still some comparisons are in order. Here are pointers to certain relevant studies on this topic.

1. Loh et al. compare the following networks on the basis of their diameter, cost-effectiveness, and the ability to emulate other networks: n-cube, Gaussian cube, reduced hypercube, and exchanged hypercube.
2. Li et al. [14] present a comparison table that details the degree, diameter, average distance, and bisection width of the following: hypercube, dual-cube, quad-cube, and the more general metacube.
3. Li et al. [15] present a comparison table that details the number of nodes, degree, diameter, and cost ratio of the following: p-ary three-cube, n-cube, cube-connected cycle, dual-cube, WK-recursive network, and recursive dual-net.

1.6 A brief commentary on certain claims by other authors

1. Some authors [10, 14] stated without a proof that the dual-cube is vertex-transitive. This paper fills up the gap.
2. Loh et al. (resp. Klavžar and Ma [10]) discuss the shortest distance between two nodes in the exchanged hypercube (resp. average node distance of the exchanged hypercube), from which the corresponding measures in the dual-cube follow. However, the proofs in this paper are more constructive and intuitive.
3. Contrary to another claim [14], the dual-cube is not edge-transitive. See Fact 2.1.

1.7 What follows

Section 2 establishes vertex transitivity of the dual-cube, while Sect. 3 derives a formula for the (shortest) distance between Node 0 and a given node z of the graph. (Vertex transitivity ensures an easy generalization of the formula.) Section 3.2 presents a description of the distance-wise node distribution of the graph that leads to an exact formula for the average node distance. Later, Sect. 4 shows that DQ_m admits an elegant hierarchical structure. The paper ends with certain concluding remarks in Sect. 5.

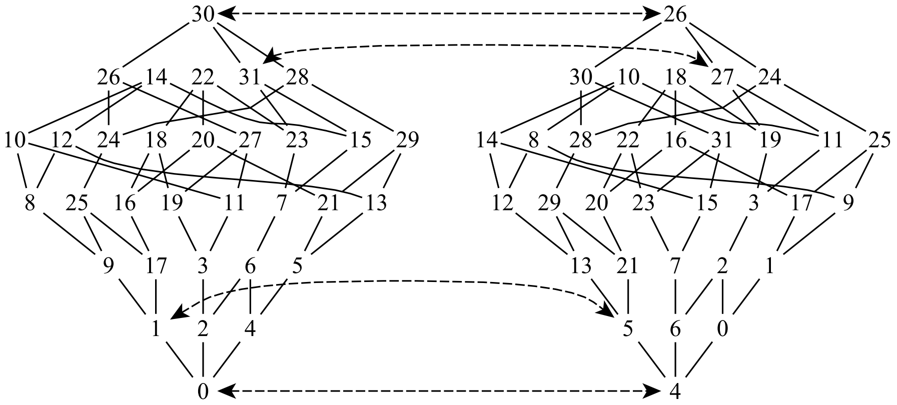

Fig. 3 An automorphism on DQ_2, ($4 \leftrightarrow 0$), vide Lemma 2.1

2 Vertex transitivity

Lemma 2.1 DQ_m *admits an automorphism that carries a given node of Type* 0 *to Node* 0.

Proof Let $z = uv0$ (binary), where $|u| = |v| = m$, and consider the mapping $\phi_z : V(\mathrm{DQ}_m) \to V(\mathrm{DQ}_m)$ given by $pqa \mapsto (p \veebar u)(q \veebar v)a$, where $|p| = |q| = m$ and $a \in \{0, 1\}$. It is easy to see that ϕ_z is well defined and $\phi_z(z) = 0$ (decimal). Further, $\phi_z(x) = y$ iff $\phi_z(y) = x$, so the mapping is total and invertible, hence a bijection.

To prove that the adjacency (resp. non-adjacency) is preserved under the mapping, first consider a node of Type 0, say $pq0$, where $|p| = |q| = m$. Whereas one of its neighbors is $pq1$, the others are $p\,q^{(0)}\,0$, $p\,q^{(1)}\,0$, …, $p\,q^{(m-1)}\,0$. Note that $\phi_z(pq0) = (p \veebar u)(q \veebar v)0$, and further observe that

(a) $\phi_z(pq1) = (p \veebar u)(q \veebar v)1$ that is adjacent to $\phi_z(pq0)$ via a cross edge, and
(b) $\phi_z\big(p\,q^{(j)}\,0\big) = (p \veebar u)(q^{(j)} \veebar v)0$ that is adjacent to $\phi_z(pq0)$ via an edge of Type 0, whre $0 \le j \le m - 1$.

By a similar argument, an analogous conclusion is reachable with respect to a node of Type 1.

Finally, let x and y be nonadjacent. If $\phi_z(x)$ and $\phi_z(y)$ were adjacent, then so must be $\phi_z(\phi_z(x))$ and $\phi_z(\phi_z(y))$. However, $\phi_z(\phi_z(x)) = x$ and $\phi_z(\phi_z(y)) = y$, a contradiction. □

Figure 3 illustrates the proof of Lemma 2.1, where $m = 2$ and $z = 4$. (The i-th node in a particular row to the left maps to the respective node on the same row to the right.

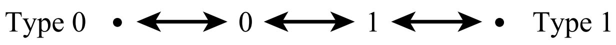

Fig. 4 Automorphisms that take a node of one type to one of another

Lemma 2.2 DQ$_m$ *admits an automorphism that carries a given node of Type* 1 *to Node* 1.

Proof Similar to that of Lemma 2.1. □

Lemma 2.3 DQ$_m$ *admits an automorphism that carries Node* 1 *to Node* 0.

Proof Let $\phi : V(\mathrm{DQ}_m) \to V(\mathrm{DQ}_m)$ be given by $uva \mapsto vu\bar{a}$, where $|u| = |v| = m$ and $a \in \{0, 1\}$. It is clear that the mapping is well-defined. Further, $\phi(y) = z$ iff $\phi(z) = y$, which means that it is a bijection. Notice that $\phi(1) = 0$ (decimal).

Let $z_1, z_2 \in V(\mathrm{DQ}_m)$, and let $z_1 = u_1v_1a_1$ and $z_2 = u_2v_2a_2$, where $|u_1| = |v_1| = |u_2| = |v_2| = m$, and $|a_1| = |a_2| = 1$, so $\phi(z_1) = v_1u_1\bar{a}_1$ and $\phi(z_2) = v_2u_2\bar{a}_2$.

1. First suppose that z_1 and z_2 are adjacent.
   (a) Let z_1 and z_2 be adjacent via a cross edge, so $u_1 = u_2$, $v_1 = v_2$ and $a_2 = \bar{a}_1$. In this case, $\phi(z_1) = v_1u_1\bar{a}_1$ and $\phi(z_2) = v_1u_1a_1$ that are clearly adjacent via a cross edge.
   (b) Let z_1 and z_2 be both of Type 0 and adjacent, so $z_1 = u_1v_10$ and $z_2 = u_1\, v_1^{(j)}\, 0$, where $0 \leq j \leq m - 1$. In this case, $\phi(z_1) = v_1u_11$ and $\phi(z_2) = v_1^{(j)}\, u_1\, 1$. It is clear that $\phi(z_1)$ and $\phi(z_2)$ are adjacent. Similarly, if z_1 and z_2 be both of Type 1 and adjacent, then $\phi(z_1)$ and $\phi(z_2)$ are adjacent.
2. Next suppose that z_1 and z_2 are non-adjacent. If $\phi(z_1)$ and $\phi(z_2)$ were adjacent, then so must be $\phi(\phi(z_1))$ and $\phi(\phi(z_2))$. However, $\phi(\phi(z_1)) = z_1$ and $\phi(\phi(z_2)) = z_2$, a contradiction. □

Remark It is easy to compute the inverses of the foregoing automorphisms.

Theorem 2.4 DQ$_m$ *is a vertex-transitive graph.*

Proof Let $y, z \in V(\mathrm{DQ}_m)$, where $y \neq z$. An automorphism that takes y to z is obtainable by using the constructions in Lemmas 2.1, 2.2, and 2.3, as follows.

1. If y and z are the same type, say t, then compose the automorphism that takes y to t with the one that takes t to z.

Table 1 Distance between Node 0 and a given node z in DQ_m

Case	Node z	Type of z	Predicate	dist(0,z)
1(a)	$0^m v0$	0	–	$\lvert v\rvert_1$
1(b)	$uv0$	0	$\lvert u\rvert_1 > 0$	$\lvert u\rvert_1 + v\rvert_1 + 2$
2	$uv1$	1	—	$\lvert u\rvert_1 + \lvert v\rvert_1 + 1$

$|u| = |v| = m$

$|u|_1$ stands for the number of 1's in the binary string u

2. If y and z are of different types, then build an automorphism that takes y to z as per the schematic that appears in Fig. 4. □

Fact 2.1 DQ_m *is not edge-transitive. For example, observe from Fig.* 3*i that the edge* $\{0, 2\}$ *of* DQ_2 *lies on a four-cycle, whereas the edge* $\{0, 1\}$ *does not lie on any four-cycle.*

3 Distance metric

Theorem 3.1 *Table* 1 *presents the distance between Node* 0 *and a givn node* z *in* DQ_m.

Proof See Algorithm 1 that traces a shortest path between two nodes in DQ_m. Check to see that, in each case, any other path is at least as long. □

Corollary 3.2 $\mathrm{dia}(\mathrm{DQ}_m) = 2m + 2$.

Proof By Table 1, there is exactly one node, viz., 1^m1^m0 arising out of Case 1(b) that is at the distance of $2m + 2$ from Node 0. Indeed, $2m + 2$ is the largest such integer. This fact and the vertex transitivity of the graph together ensure that $\mathrm{dia}(\mathrm{DQ}_m) = 2m + 2$. □

Note that DQ_m is a spanning subgraph of Q_{2m+1}, having about 50% fewer edges, yet its diameter is practically equal to that of the latter.

Algorithm 1 Tracing a shortest path between two nodes in DQ_m

```
Require: $m \geq 1$, and two nodes $x$ and $y$ (binary), where $|x| = |y| = 2m+1$
let $\phi_x$ be the automorphism that maps $x$ to Node 0 ($0^{2m+1}$ in binary)
▷ make use of the constructions in the proofs of Lemmas 2.1 − 2.3 for the foregoing purpose.
let $\phi_x(y) = z = uvc$ (say) where $|u| = |v| = m$, and $|c| = 1$;
▷ $\text{dist}(x, y) = \text{dist}(0, z)$
▷ let $|v|_1$ stand for the number of 1's in the binary string $v$
if ($|u|_1 = 0$ and $c = 0$) then
    return the path $0^m0^m0 \overset{|v|_1}{\rightsquigarrow} 0^mv0$;
else if ($|u|_1 > 0$ and $c = 0$) then
    return the path $0^m0^m0 \overset{|v|_1}{\rightsquigarrow} 0^mv0 \xrightarrow{1} 0^mv1 \overset{|u|_1}{\rightsquigarrow} uv1 \xrightarrow{1} uv0$;
else                                                                ▷ $c = 1$
    return the path $0^m0^m0 \overset{|v|_1}{\rightsquigarrow} 0^mv0 \xrightarrow{1} 0^mv1 \overset{|u|_1}{\rightsquigarrow} uv1$;
end if
```

3.1 Distance-wise node distribution

For $1 \leq d \leq 2m + 2$, let n_d denote the number of nodes at a distance of d from Node 0 in DQ$_m$, and let q_d denote the number of nodes at a distance of $d - 1$ from Node 0 in Q_{2m+1}. It is well known that $q_d = C(2m + 1, d - 1)$.

Theorem 3.3 *If* $m \geq 1$, *then* $n_d = C(2m + 1, d - 1) + C(m, d) - C(m, d - 2)$.

Proof Refer to Table 1. Whereas Case 1(a) contributes $C(m, d)$ nodes, Case 1(b) contributes $C(2m, d - 2) - C(m, d - 2)$ nodes, where $C(m, d - 2)$ denotes the number of nodes of the form 0^mv0, already covered in Case 1(a). Finally, Case 2 contributes $C(2m, d - 1)$ nodes. The claim is immediate. □

Table 2 exemplifies Theorem 3.3 for $m = 10$, whereas Fig. 5 depicts it. Not surprisingly, n_d achieves its maximum at $d = m + 2$. Meanwhile, let diff $= n_d - q_d = C(m, d) - C(m, d - 2)$.

Table 2 DQ_m vs. Q_{2m+1}, $m = 10$

d	n_d	q_d	diff $= n_d - q_d$
$2m + 2 = 22$	1	1	0
21	21	21	0
20	210	210	0
19	1330	1330	0
18	5985	5985	0
17	20,349	20,349	0
16	54,264	54,264	0
15	116,280	116,280	0
14	203,490	203,490	0
$m + 3 = 13$	293,930	293,930	0
$m + 2 = 12$	352,715	352,716	-1
11	352,706	352,716	– 10
10	293,886	293,930	– 44
9	203,380	203,490	– 110
8	11,615	116,280	– 165
7	54,132	54,264	– 132
6	20,349	20,349	0
5	6117	5985	132
4	1495	1330	165
3	320	210	110
2	65	21	44
1	11	1	10
0	1	0	1

Number of nodes in DQ_{10} : $2^{21} = 2,097,152$

Number of nodes in Q_{21} : $2^{21} = 2,097,152$

3.2 Average node distance of DQ_m

Table 3 computes the average node distance of DQ_m, i.e., $\frac{1}{2^{2m+1}} \sum_{d=0}^{2m+2} d \cdot n_d$ $= \frac{1}{2^{2m+1}} \sum_{d=0}^{2m+2} \Big(d \cdot C(2m+1, d-1) + d \cdot C(m, d) - d \cdot C(m, d-2) \Big)$.

Notice that the average node distance of DQ_m is practically equal to that of Q_{2m+1}, viz., $m + \frac{1}{2}$.

3.3 DQ_m versus Q_{2m+1}

It turns out that the distance-wise node distribution of DQ_m closely parallels that of Q_{2m+1}. To that end, recall that diff $= n_d - q_d = C(m, d) - C(m, d-2)$.

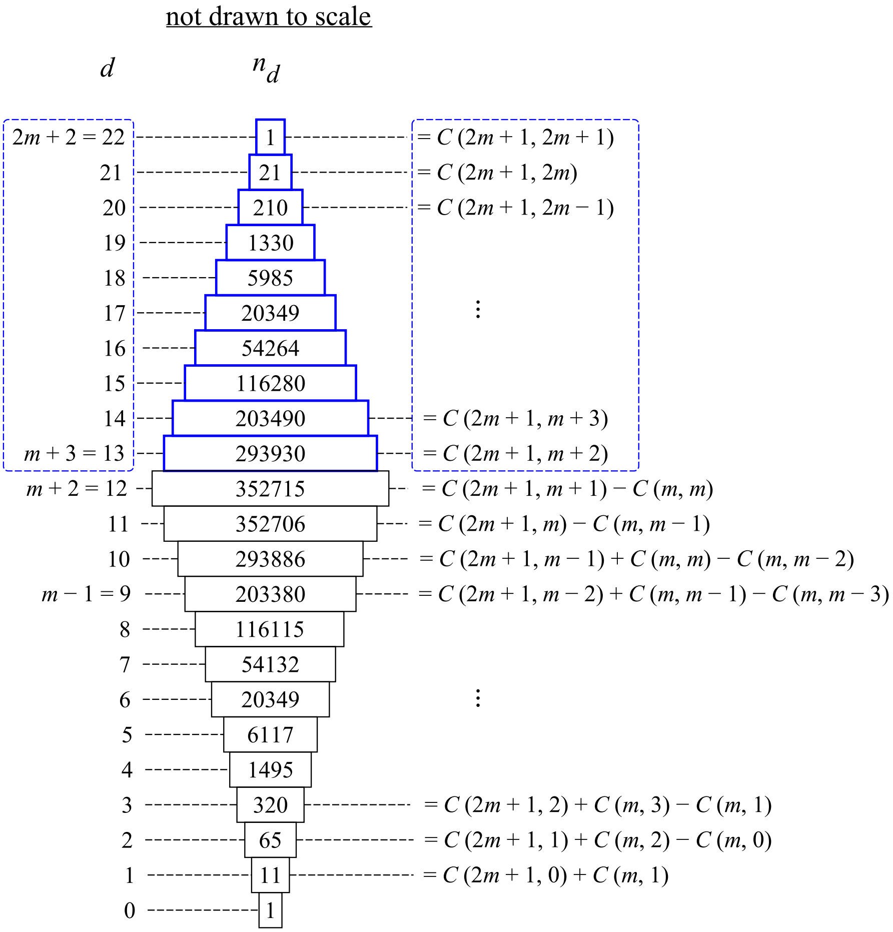
not drawn to scale
d
n_d
$2m+2=22$ 1 $=C(2m+1,2m+1)$
21 21 $=C(2m+1,2m)$
20 210 $=C(2m+1,2m-1)$
19 1330
18 5985
17 20349 ⋮
16 54264
15 116280
14 203490 $=C(2m+1,m+3)$
$m+3=13$ 293930 $=C(2m+1,m+2)$
$m+2=12$ 352715 $=C(2m+1,m+1)-C(m,m)$
11 352706 $=C(2m+1,m)-C(m,m-1)$
10 293886 $=C(2m+1,m-1)+C(m,m)-C(m,m-2)$
$m-1=9$ 203380 $=C(2m+1,m-2)+C(m,m-1)-C(m,m-3)$
8 116115
7 54132
6 20349 ⋮
5 6117
4 1495
3 320 $=C(2m+1,2)+C(m,3)-C(m,1)$
2 65 $=C(2m+1,1)+C(m,2)-C(m,0)$
1 11 $=C(2m+1,0)+C(m,1)$
0 1

Fig. 5 Distance-wise node distribution of DQ_{10}

The following points are relevant.

1. If $2m+2 \geq d \geq m+3$, then diff $=0$..
2. If $m+2 \geq d \geq \frac{1}{2}(m+2)$, then diff is negative.
3. If $\frac{1}{2}(m+2) \geq d \geq 1$, then diff is positive.
4. For $m+2 \geq d \geq 1$, diff is symmetric about the point $d=\frac{1}{2}(m+2)$, and its absolute value is a very small percentage of $C(2m+1,d-1)$.

Figure 6 presents the trace of diff vs. d for $m=10$.

Table 3 Computing the average node distance

$\sum_{d=1}^{2m+2}\big(d\cdot C(2m+1,d-1)\big)$	=	$\sum_{i=0}^{2m+1}\big((i+1)\cdot C(2m+1,i)\big)$
	=	$(2m+1)2^{2m}+2^{2m+1}$
$\sum_{d=0}^{m}\big(d\cdot C(m,d)\big)$	=	$m2^{m-1}$
$-\sum_{d=2}^{m+2}\big(d\cdot C(m,d-2)\big)$	=	$-\sum_{i=0}^{m}\big((i+2)\cdot C(m,i)\big)$
	=	$-(m2^{m-1}+2^{m+1})$
Algebraic sum	=	$(m+1)\,2^{2m+1}+2^{2m}-2^{m+1}$
Average node distance	=	$\frac{(m+1)\,2^{2m+1}+2^{2m}-2^{m+1}}{2^{2m+1}}$
	=	$m+\frac{3}{2}-2^{-m}$
Identities used: $\sum_{i=0}^{n}C(n,i)=2^n$ and $\sum_{i=0}^{n}i\,C(n,i)=n2^{n-1}$		

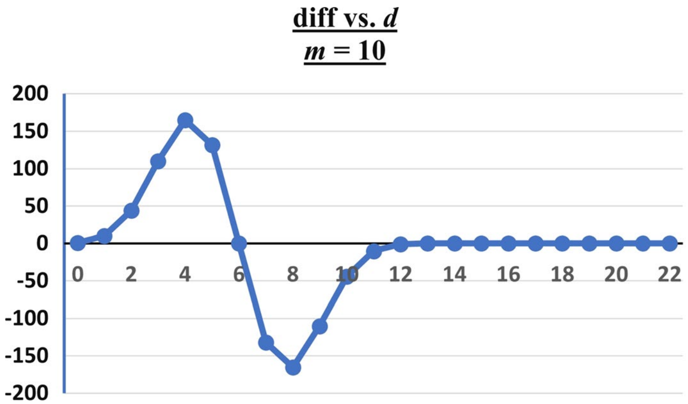

Fig. 6 diff vs. d for $m=10$, $0\le d\le 2m+2=22$, vide Table 2

4 Hierarchical structure of DQ_m

It turns out that DQ_m admits an interesting hierarchical structure. In particular, there exists a sequence of graphs $\langle G_0,\dots,G_{2m}\rangle$ such that (1) $G_0=K_2$, (2) G_{i+1} is obtainable from G_i by introducing (not a perfect) matching between two copies of G_i, $0\le i\le 2m-1$, and (3) G_{2m} is isomorphic to DQ_m. See Algorithm 2.

Algorithm 2 Constructing the sequence of graphs $\langle G_1, \ldots, G_{2m}\rangle$

```
Require: $m \geq 1$
 1: let $G_0$ be the two-node graph $0 - 1$;
 2: for ($i = 0$ to $m - 1$) do
 3:     create two copies of $G_i$;
 4:     add $2^{i+1}$ to each node in the second copy;
 5:     for (each node $u$ of Type 0 in the first copy) do
 6:         introduce the edge $\{u, u + 2^{i+1}\}$;
 7:     end for
 8:     call the resulting graph as $G_{i+1}$;
 9: end for
10: ▷
11: for ($i = m$ to $2m - 1$) do
12:     create two copies of $G_i$;
13:     add $2^{i+1}$ to each node in the second copy
14:     for (each node $u$ of Type 1 in the first copy) do
15:         introduce the edge $\{u, u + 2^{i+1}\}$;
16:     end for
17:     call the resulting graph as $G_{i+1}$;
18: end for
19: return the sequence of graphs $\langle G_1, \ldots, G_{2m}\rangle$;
```

It is easy to see that each statement of the algorithm is well-defined. See Fig. 7 for an illustration, where $m = 2$.

Lemma 4.1 *For $m \geq 1$, let $G_1, \ldots, G_{2m}$ be the sequence of graphs returned by Algorithm 2, and let $1 \leq i \leq 2m$. Then*

1. *G_i is a bipartite graph, where each partite set is of the same cardinality*
2. *Each partite set of G_i has as many nodes of Tyoe 0 as those of Type 1*
3. $V(G_i) = \{0, \ldots, 2^{i+1} - 1\}$
4. *Each edge of G_i is valid relative to* DQ_m
5. $|E(G_i)| = (i + 2)2^{i-1}$.

Proof Use induction on i. To that end, first check to see that the claim holds with respect to G_1 that is the graph $1 - 0 - 2 - 3$. The induction step follows.

1. Let V_0 and V_1 be the two partite sets of G_i, so $V_0 + 2^{i+1}$ and $V_1 + 2^{i+1}$ are the two partite sets of of the second copy of G_i immediately after the execution of Step 4 (resp. Step 13) of the algorithm.

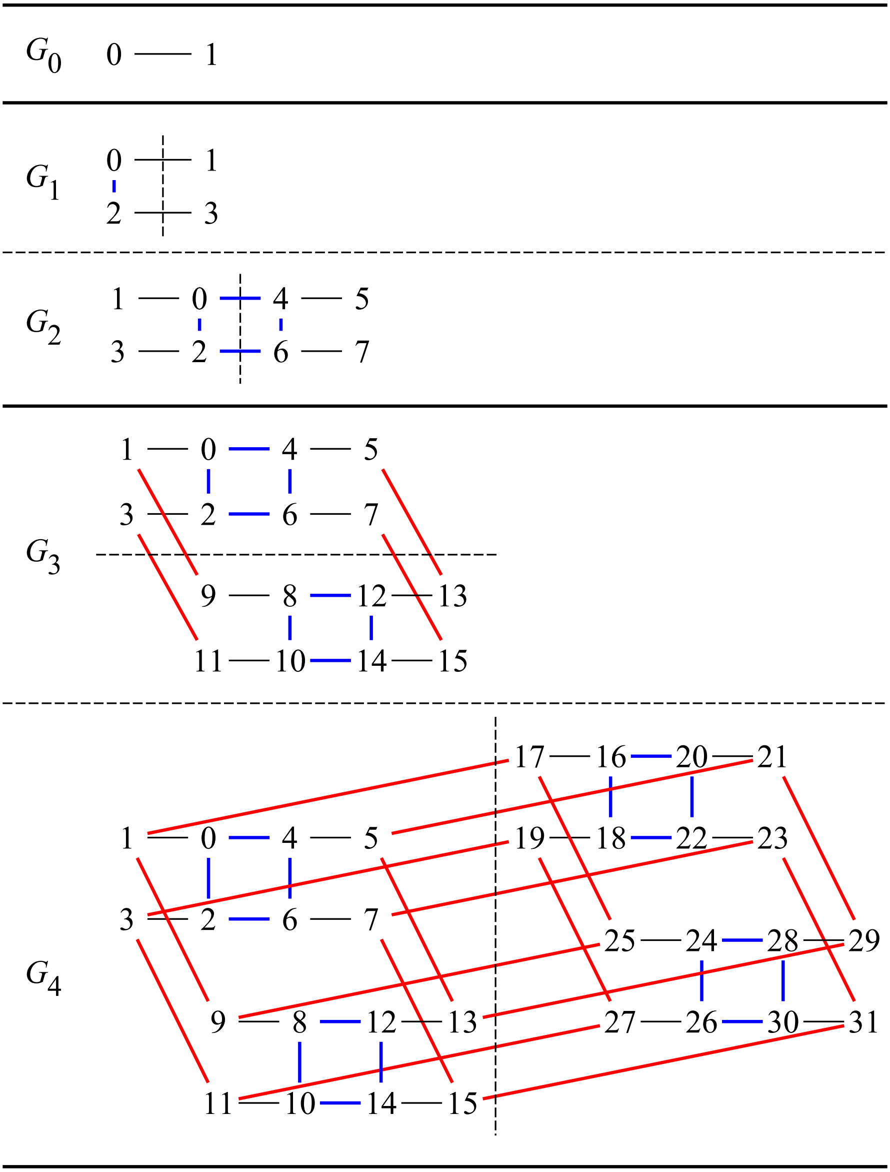

Fig. 7 Hierarchical structure of DQ_2

Every edge introduced during execution of each of the two "inner for" loops runs between V_0 and $V_0 + 2^{i+1}$, or between V_1 and $V_1 + 2^{i+1}$. In that light, G_{i+1} is a bipartite graph with $V_0 \cup (V_1 + 2^{i+1})$ and $V_1 \cup (V_0 + 2^{i+1})$ as the two partite sets. Further, the hypothesis that nodes of the two types are equinumerous in each of V_0 and V_1 ensures that this is the case with respect to each of

$V_0 \cup (V_1 + 2^{i+1})$ and $V_1 \cup (V_0 + 2^{i+1})$ as well. Simultaneously, $|V_0| = |V_1|$ ensures that $|V_0 \cup (V_1 + 2^{i+1})| = |V_1 \cup (V_0 + 2^{i+1})| = 2^{i+1}$.

The foregoing argument also shows that $V(G_{i+1}) = \{0, \dots, 2^{i+2} - 1\}$.

2. (a) Let $\{x, y\} \in E(G_i)$. By induction hypothesis, $|x| = |y| = i + 1$, where x and y are in binary. (In decimal, $0 \le x, y \le 2^{i+1} - 1$.) Also by induction hypothesis, $\{x, y\}$ is a valid edge relative to DQ_m. In that light, the edge $\{1x, 1y\}$, which appears in the second copy of G_i, is valid relative to DQ_m. Indeed, if $\{x, y\}$ is a cross edge (resp. an edge of Type 0, or an edge of Type 1), then so is $\{1x, 1y\}$. (In decimal, $1x = 2^{i+1} + x$ and $1y = 2^{i+1} + y$.)
   (b) If $1 \le i \le m - 1$, then each edge introduced during execution of the "first inner for" loop runs between two nodes of the form $0^m x0$ and $0^m y0$ (binary) that are of Type 0, where $|x| = |y| = m$ and $\mathrm{H}(x, y) = 1$. Analogously, if $m \le i \le 2m - 1$, then each edge introduced during execution of the "second inner for" loop runs between nodes of the form $ux1$ and $vx1$ that are of Type 1, where $|u| = |v| = |x| = m$ and $\mathrm{H}(u, v) = 1$. In either case, all such edges are valid relative to DQ_m.

   It follows that the edges that appear in G_{i+1} are valid relative to DQ_m.
3. Note that $|E(G_1)| = 3$, and $|E(G_{i+1})| = 2|E(G_i)| + 2^i$, where $i \ge 1$. (Each partite set of G_{i+1} has 2^i nodes of each type.) Here is a solution to the recurrence: $|E(G_{i+1})| = (i + 3)2^i$.

This concludes the induction. □

Note: It is easy to see that the running time of Algorithm 2 is proportional to the sum of the sizes of the graphs $G_0, \dots, G_{2m}$.

Theorem 4.2 *The graph G_{2m} returned by Algorithm 2 is isomorphic to* DQ_m.

Proof By Lemma 4.1, (1) $V(G_{2m}) = \{0, \dots, 2^{2m+1} - 1\}$ that coincides with $V(\mathrm{DQ}_m)$, (2) each edge of G_{2m} is valid relative to DQ_m, and (3) $|E(G_{2m})| = (m + 1)2^{2m}$ that coincides with $|E(\mathrm{DQ}_m)|$. Hence the result. □

Remark The result of this section is likely to be useful in the future in deriving other results with respect to the dual-cube and probably the exchanged hypercubes. This is because the sequence of graphs returned by Algorithm 2 is amenable to an inductive treatment.

Interestingly, it is easy to construct an algorithm that takes G_{2m} as the input and performs a sequence of "inverse" operations, as follows: At the i-th step, partition G_{2m-i} into two subgraphs, each isomorphic to G_{2m-i-1}, $0 \le i \le 2m - 1$, where $G_0, \dots, G_{2m}$ are as in Algorithm 2.

5 Concluding remarks

The dual-cube DQ_m is a regular, connected spanning subgraph of the hypercube Q_{2m+1} [14]. It admits a number of welcome properties that make it amenable to an application as a network topology, especially for systems with large node sizes [8, 11, 12, 19, 22]. Results in this paper significantly enhance the importance of the network. Further, the techniques employed are likely to pave the way for similar studies on the quad-cube and the oct-cube that belong to the greater family of such graphs, called the metacube [14].

It is conceivable that there exists a closer relationship between the exchanged hypercubes and the dual-cube. Algorithm 2 is likely to be extended for that purpose. Another relevant question is to determine the smallest n such that DQ_m is a vertex-induced subgraph of Q_n.

Acknowledgements The author thanks the anonymous referees for their comments on the earlier draft that led to an overall improvement in the presentation of the paper.

References

1. Angjeli A, Cheng E, Lipták L (2013) Linearly many faults in dual-cube-like networks. Theor Comput Sci 472:1–8
2. Chen J-C, Tsai C-H (2011) Conditional edge-fault-tolerant Hamiltonicity of dual-cubes. Inform Sci 181(3):620–627
3. Chen YW (2007) A comment on 'The exchanged hypercube'. IEEE Trans Parallel Distrib Syst 18:576
4. Chen S-Y, Kao S-S (2010) Hamiltonian connectivity and globally 3*-connectivity of dual-cube-extensive networks. Comput Elect Eng 36(3):404–413
5. Fan W, Fan J, Lin CK (2019) An efficient algorithm for embedding exchanged hypercubes into grids. J Supercomput 75:783–807
6. Hayes JP, Mudge TN (1989) Hypercube supercomputers. Proc IEEE 17(12):1829–1841
7. Jha PK (2015) A comment on 'The domination number of exchanged hypercubes'. Inf Process Lett 115(2):343–344
8. Jha PK (2015) 1-perfect codes over dual-cubes vis-à-vis Hamming codes over hypercubes. IEEE Trans Inf Theory 61(8):4259–4268
9. Klavžar S, Ma M (2014) The domination number of exchanged hypercubes. Inf Process Lett 114(4):159–162
10. Klavžar S, Ma M (2014) Average distance, surface area, and other structural properties of exchanged hypercubes. J Supercomput 69:306–317
11. Lai C-J, Tsai C-H (2008) On embedding cycles into faulty dual-cubes. Inf Process Lett 109:147–150
12. Li Y, Peng S, Chu W (2004) Efficient collective communications in dual-cube. J Supercomput 28:71–90
13. Li Y, Peng S, Chu W (2005) Fault-tolerant cycle embedding in dual-cube with node faults. Int J High Perf Comput Netw 3(1):45–53
14. Li Y, Peng S, Chu W (2010) Metacube − a versatile family of interconnection networks for extremely large-scale supercomputers. J Supercomput 53(2):329–351

15. Li Y, Peng S, Chu W (2011) Longest-paths and fault-tolerant routing on recursive dual-net. Int J FOCS 22(5):1001–1018
16. Loh PKK, Hsu WJ, Pan Y (2005) The exchanged hypercube. IEEE Trans Parallel Distrib Syst 16:866–874
17. Ma M, Liu B (2010) Cycles embedding in exchanged hypercubes. Inf Process Lett 110(2):71–76
18. Pai K-J, Chang J-M (2016) Constructing two completely independent spanning trees in hypercube-variant networks. Theor Comp Sci 652:28–37
19. Shih Y-K, Chuang H-C, Kao S-S, Tan JJM (2010) Mutually independent Hamiltonian cycles in dual-cubes. J Supercomput 54:239–251
20. Saad Y, Schultz H (1988) Topological properties of hypercubes. IEEE Trans Comput 37(7):867–872
21. West DB (2001) Introduction to graph theory, 2nd edn. Prentice-Hall, Englewood Cliffs, NJ, USA
22. Wu C, Wu J (2003) On self-similarity and Hamiltonicity of dual-cubes. Proc Int Parallel Distrib Processes Symp IEEE (7 pages)
23. Zhou S, Chen L, Xu J-M (2012) Conditional fault diagnosability of dual-cubes. Int J FOCS 23(8):1729–1747
24. Zhang Q, Xu L, Zhou S, Yang W (2020) Reliability analysis of subsystem in dual cubes. Theor Comp Sci 816:249–259
25. Zhao S-L, Hao R-X (2019) The generalized 4-connectivity of exchanged hypercubes. Appl Math Comp 347:342–353