International Journal of Foundations of Computer Science

LINEAR LAYOUT OF GENERALIZED HYPERCUBES*

KOJI NAKANO
School of Information Science, Japan Advanced Institute of Science and Technology
Tatsunokuchi, Ishikawa 923-1292, Japan

Received (received date)
Revised (revised date)
Communicated by Editor's name

ABSTRACT

This paper deals with two kinds of generalized hypercubes: a d-dimensional c-ary clique C_c^d and a d-dimensional c-ary array A_c^d. A d-dimensional c-ary clique C_c^d has nodes labeled by c^d integers from 0 to c^d-1 and two nodes are connected by an edge if and only if the c-ary representations of their labels differ by one and only one digit. A d-dimensional c-ary array A_c^d also has nodes labeled by c^d integers from 0 to c^d-1, and two nodes are connected if and only if the c-ary representations of their labels differ by one and only one digit and the absolute value of the difference in that digit is 1. Further, an n-node c-ary clique $C_c^{(n)}$ is the induced subgraph of C_c^d ($n \geq c^d$) with nodes labeled by integers from 0 to $n-1$. The main contribution of this paper is to clarify several topological properties of A_c^d and C_c^d in terms of their linear layouts. For this purpose, we first prove that $C_c^{(n)}$ is a maximum subgraph of $C_c^{(m)}$, that is, $C_c^{(n)}$ has the largest number of edges over all n-node subgraphs of $C_c^{(m)}$, whenever $n \leq m$. Using this fact, we show the exact values of the bisection width, cut width, and total edge length of C_c^d. We also show the exact value of the bisection width of A_c^d and nearly tight values of the cut width and the total edge length of A_c^d.

Keywords: Linear Layout; Generalized Hypercubes; Cut width; Bisection width; Total edge length; Maximum subgraphs

1. Introduction

A d-dimensional hypercube is a graph with 2^d nodes and $d2^{d-1}$ edges. The nodes are labeled by 2^d integers from 0 to 2^d-1, and they are connected by edges if and only if the binary representations of their labels differ by one and only one bit. Processor networks based on the hypercube topology can solve many problems efficiently [1, 2, 15] and they are robust against faults [5]. Many parallel machines based on the hypercube topology have been studied. However, the topology of the hypercube is not flexible in the sense that, for any given number of processors, the number of links required to connect processors based on the hypercube topology is fixed. Thus, if insufficient links are available, we must compromise on a small

* Preliminarily version appeared in Proc. of Workshop on Graph-Theoretic Concepts in Computer Science [19]

hypercube network. On the other hand, if more links are available, we cannot take advantage of improve the communication capability if the topology is based on the hypercubes. Therefore, generalized hypercubes offers more flexible structure, are used instead.

This paper focuses on two types of generalized hypercubes defined as follows. *A d-dimensional c-ary clique* C_c^d has nodes labeled by c^d integers from 0 to c^d-1. The nodes are connected by the edges if and only if the c-ary representations of their labels differ by one and only one digit (Fig. 1). Graph C_c^d is regular of degree $(c-1)d$ and has c^d nodes, $(c-1)dc^d/2$ edges, and a diameter of d. Let K_c be a c-node clique and L_c be a c-node linear array. It is not difficult to confirm that C_c^d is the Cartesian product of d c-node cliques K_c. *A d-dimensional c-ary array* A_c^d also has nodes labeled by c^d integers from 0 to c^d-1. The nodes are connected if and only if the c-ary representations of their labels differ by one and only one digit and the absolute value of the difference in that digit is 1. (Fig. 2). Graph A_c^d has c^d nodes, $(c-1)dc^{d-1}$ edges, and a diameter of $d(c-1)$. Again, it is not difficult to see that A_c^d is the Cartesian product of d c-node linear arrays L_c. Graphs A_c^d and C_c^d have similar topologies: each side of A_c^d is L_c, while each side of C_c^d is K_c. Clearly, A_c^d is a subgraph of C_c^d. Similar generalizations of hypercubes have been given in [7, 9, 26]. Several algorithms on parallel computers based on C_c^d and A_c^d topologies have been shown [7, 15]. The analysis of their topological properties is important, because they are very attractive as network topologies of future parallel computers. Further, C_c^d and A_c^d include typical topologies which are used for parallel machines: C_c^1 is equal to a c-node clique. A_c^1, A_c^2, and A_c^3 correspond to a c-node linear array, a c^2-node 2-dimensional array, and a c^3-node 3-dimensional array, respectively. Both C_2^d and A_2^d are equal to a d-dimensional (binary) hypercube. Therefore, the results presented in this paper can be applied to these topologies of graphs.

For later reference, we define a class of subgraphs of C_c^d. *An n-node c-ary clique* $C_c^{(n)}$ has n nodes labeled by n integers from 0 to $n-1$ and the nodes are connected in the same way as C_c^d. Note that n is not restricted to a power of c. Clearly, $C_c^{(n)}$ is an induced subgraph of $C_c^{(m)}$ whenever $n \le m$. Figure 3 illustrates $C_4^{(14)}$.

Let $G=(V,E)$ denote a graph such that V and E are a set of nodes and a set of edges, respectively. *A linear layout of* $G=(V,E)$ is a one-to-one mapping $l : V \to \{0,1,2,\ldots,|V|-1\}$. A linear layout l of a graph means that each node $u \in V$ is positioned at coordinate $l(u)$ on the baseline. Figures 4 and 5 illustrate examples of a linear layout of A_4^2. Note that, in Figure 5, each node u $(0 \le u \le |V|-1)$ is positioned at coordinate u. We denote such the layout *the natural order layout.* Let N denote the natural order layout, that is, $N(u)=u$ for every u $(0 \le i \le |V|-1)$. Note that a linear layout can take any of the $|V|!$ permutations, not just the natural order layout.

The complexity of $G=(V,E)$ in terms of a linear layout is evaluated by the following parameters: *the (minimum) bisection width*, *the cut width*, and *the total edge length.* The definitions of these parameters are as follows. *The width of a graph G under a linear layout l at a gap i* denoted by $C(G,l,i)$ is a set of edges connecting

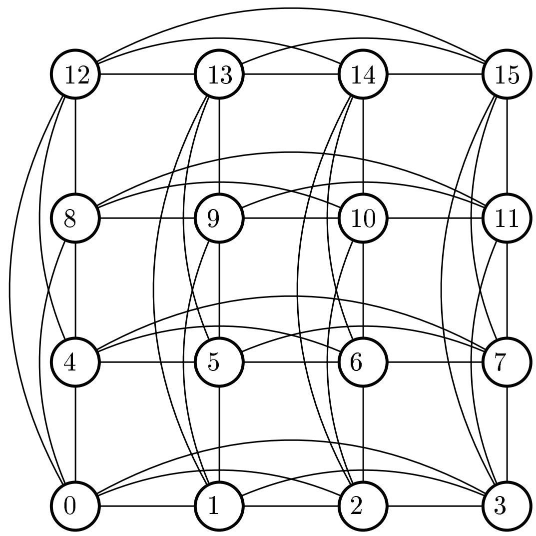

Figure 1: A 2-dimensional 4-ary clique, C_4^2

a node at a position less than i and one at a position larger than or equal to i. In other words, $C(G,l,i) = \{(u,v) \in E \mid 0 \leq l(u) < i \leq l(v) \leq |V|-1\}$. *The bisection width* $\mathrm{BW}(G)$ *of a graph* G is the minimum number of edges in $C(G,l,\lfloor |V|/2 \rfloor)$ over all linear layouts, that is,

$$\mathrm{BW}(G) \quad = \quad \min_{l} |C(G,l,\lfloor |V|/2 \rfloor)|$$

In other words, $\mathrm{BW}(G)$ is the minimum number of edges which must be removed to separate the graph into two disjoint and equal-sized subgraphs. *The cut width of a graph* G *under a linear layout* l is the maximum of $|C(G,l,i)|$ over all gaps i, *i.e.* $\max_i |C(G,l,i)|$. *The cut width* $\mathrm{CW}(G)$ *of a graph* G is the minimum cut width over all linear layouts, that is,

$$\mathrm{CW}(G) \quad = \quad \min_{l} \max_{i} |C(G,l,i)|$$

This parameter indicates the number of tracks required by the best linear layout. Let us define *the length of edge* $(u,v) \in E$ *under a linear layout* l is $|l(u) - l(v)|$. *The total edge length of a graph* G *under a linear layout* l is $\sum_{(u,v)\in E} |l(u) - l(v)|$. Further, *the total edge length* $\mathrm{TL}(G)$ *of a graph* G is defined as the minimum of this value over all linear layouts, that is,

$$\mathrm{TL}(G) \quad = \quad \min_{l} \sum_{(u,v)\in E} |l(u) - l(v)|.$$

It is not difficult to confirm that, the total edge length is equal to the total cut. In

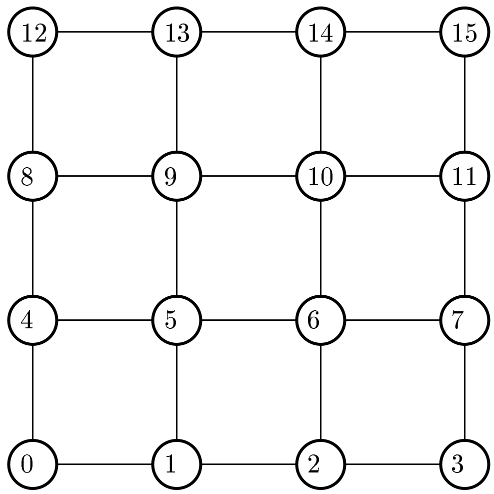

Figure 2: A 2-dimensional 4-ary array, A_4^2

other words,

$$\sum_{(u,v)\in E} |l(u) - l(v)| \quad = \quad \sum_{i=1}^{|V|-1} |C(G,l,i)|$$

holds for any graph G and linear layout l. It follows that, the total edge length of G can be computed using the following formula:

$$\mathrm{TL}(G) \quad = \quad \min_{l} \sum_{i=1}^{|V|-1} |C(G,l,i)|.$$

The main contribution of this paper is to clarify these parameters of C_c^d and A_c^d for every c and d. It is quite to compute their exact values, because they determine the lower bound of the layout area in the VLSI model. For example, the layout area of a processor network is at least $\Omega(B^2)$ if the corresponding graph has bisection width B [14, 22], and the number of tracks of a processor network in a horizontal layouts requires C layers if the corresponding graph has cut width C. The total edge length has applications in the coding theory [13] and storage management [21]: Minimizing the total edge length of generalized hypercubes corresponds to minimizing the error of a c-ary channel, and to minimizing the efficiency of managing a d-dimensional data structure in a paging environment. However, computing the exact values of them is a hard problem. For a given graph and an integer k, determining whether the bisection width of the graph is at most k is also NP-complete [12]. Similarly, the problem to determine the cut width is NP-complete even if the degree of the graph is restricted [16].

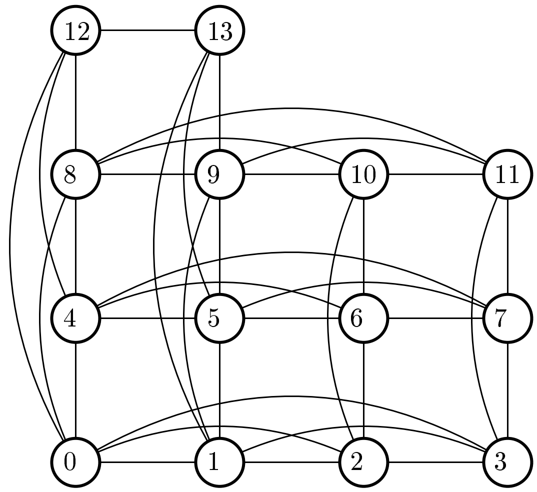

Figure 3: A 14-node 4-ary clique, $C_4^{(14)}$

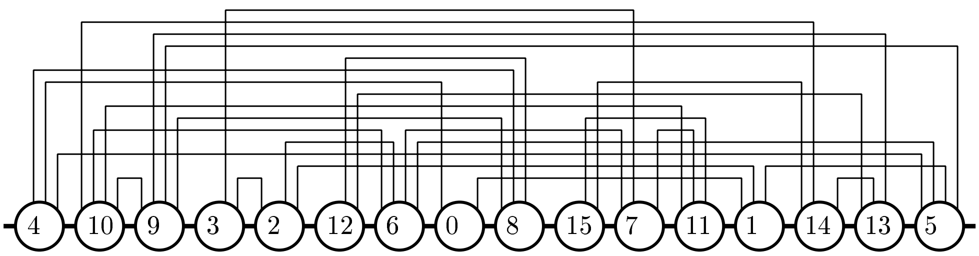

Figure 4: An example of a linear layout of A_4^2

Several articles have been devoted to the evaluation of the above parameters. Brebner [8], Manabe et al. [17], and Nakano et al. [18] have proved that the bisection width of a d-dimensional binary hypercube is 2^{d-1} using different methods. Leighton [15] showed that the bisection width of A_c^d is c^{d-1} if c is even by embedding a directed complete graph in A_c^d. Wada et al. [24] proved that the bisection width of C_c^d is $c^{d+1}/4$ if c is even in a similar way to the Leighton's proof. However, they failed to obtain the exact value of it when c is odd: the bisection width of C_c^d takes a value between $\lceil c^{d+1}/4 - 1/(4c^{d-1}) \rceil$ and $(c+1)(c^d-1)/4$ (inclusive). Nakano et al. [18] also proved that the cut width of C_2^d is $\lfloor 2^{d+1}/3 \rfloor$. Wada et al. [25] also proved that the cut width of C_c^d is at most $c^2(c^d-1)/\{4(c-1)\}$. Niepel et al. [20] showed that the total edge length of an $n \times 2$-node array is $5n-4$ and conjectured that the total edge length of an $n \times m$-node array is $n(m^2+m-1)-m^2$. Harper [13] showed that the total edge length of a d-dimensional hypercube is $2^{d-1}(2^d-1)$. DeMillo

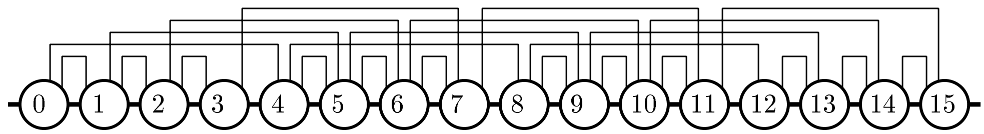

Figure 5: The natural order layout of A_4^2

Table 1: Our results and previously known results

	Bisection width	Cut width	Total edge length
binary hypercube	Brebner [8] Manabe [17] Nakano [18] exact	Nakano [18] exact	Harper [13] exact
d-dimensional c-ary clique C_c^d	Wada [24] exact when c is even	Wada [25] only upper bound	
	This paper exact	This paper exact	This paper exact
d-dimensional c-ary array A_c^d	Leighton [15] exact when c is even		DeMillo [10] lower bound when $d = 2$
	This paper exact	This paper nearly exact	This paper nearly exact

et al. [10] showed that the total edge length of a 2-dimensional hypercube is at least $n^3/6$.

In this paper, we will evaluate the bisection width, cut width, and total edge length of both C_c^d and A_c^d. In Section 2, we consider how many edges a subgraph of $C_c^{(m)}$ with n ($n \le m$) nodes may have, and show that $C_c^{(n)}$ has the largest number of edges of all subgraphs with n nodes. In other words, $C_c^{(n)}$ is the maximum subgraph of $C_c^{(m)}$ if $n \le m$. Section 3 uses this fact to get the exact values of the bisection width, cut width, and total edge length of C_c^d. Section 4 presents a method for converting C_c^d into A_c^d and get the exact value of the bisection width of A_c^d, and nearly exact values of the cut width and the total edge length of A_c^d. Table 1 shows the results obtained in this work and the previously known results.

This paper organized as follows: In Section 2, we prove that $C_c^{(n)}$ is a maximum subgraph of $C_c^{(m)}$ whenever $n \le m$. The proof shown in Section 2 is simpler than preliminary version of this paper[19]. Using this fact, Section 3 shows the exact values of $\mathrm{BW}(C_c^d)$, $\mathrm{CW}(C_c^d)$, and $\mathrm{TL}(C_c^d)$. In Section 4, we show the upper bounds and the lower bounds of $\mathrm{BW}(A_c^d)$, $\mathrm{CW}(A_c^d)$, and $\mathrm{TL}(A_c^d)$, respectively. Section 5 offers concluding remarks and mentions several related results that have been obtained after the preliminary version of this paper[19] was published.

2. A maximum subgraph of C_c^d

The main purpose of this section is to prove the following theorem:

Theorem 1 *$C_c^{(n)}$ is a maximum subgraph of $C_c^{(m)}$ whenever $n \leq m$.*

Theorem 1 can be proved by the following three lemmas.

Lemma 1 *Let f_c be the function defined as follows:*

$$f_c(n) = \begin{cases} n(n-1)/2 & \text{if } n \leq c, \\ \sum_{i=0}^{c-1} \{f_c(\lfloor (n+i)/c \rfloor) + (c-i-1)\lfloor (n+i)/c \rfloor\} & \text{if } n > c. \end{cases}$$

For all $n \geq 1$, $C_c^{(n)}$ has exactly $f_c(n)$ edges.

Lemma 2 *Let g_c be the function defined as follows:*

$$g_c(n) = \begin{cases} n(n-1)/2 & \text{if } n \leq c, \\ \max\{\sum_{i=0}^{c-1} \{g_c(n_i) + (c-i-1)n_i\} \mid \\ \qquad n_0 \leq n_1 \leq \cdots \leq n_{c-1} < n = \sum_{i=0}^{c-1} n_i\} & \text{if } n > c. \end{cases}$$

For any subgraph $G = (V, E)$ of $C_c^{(m)}$, $|E| \leq g_c(|V|)$ always holds.

Lemma 3 *For every c and n, $f_c(n) = g_c(n)$ always holds.*

Note that the division of an integer n into the same c values as equally as possible can be represented as

$$\lfloor n/c \rfloor, \lfloor (n+1)/c \rfloor, \lfloor (n+2)/c \rfloor, \ldots, \lfloor (n+c-1)/c \rfloor.$$

In fact, the sequence is $c - r$ q's followed by r $(q+1)$'s where $n = q \cdot c + r$ $(0 \leq r \leq c-1)$. Thus, $g_c(n)$ is evaluated by computing the maximum over all divisions of n, while $f_c(n)$ is evaluated for the equal-sized division of n. It follows that $f_c(n) \leq g_c(n)$ for every c and n. However, Lemma 3 claims $f_c(n) = g_c(n)$.

Lemma 1 shows the number of edges of $C_c^{(n)}$, and Lemma 2 shows the upper bound of the number of edges of the maximum subgraph of C_c Hence, from Lemma 3, the number of edges of $C_c^{(n)}$ is equal to the number of edges of the maximum subgraph with n nodes. Therefore, these lemmas imply Theorem 1. In order to complete the proof of Theorem 1, we prove the three lemmas in this section.

For later reference, let $V_c^{(n)}$ and $E_c^{(n)}$ denote the sets of nodes and edges of $C_c^{(n)}$, respectively. In other words,

$$\begin{aligned} V_c^{(n)} &= \{0, 1, \ldots, n-1\}, \\ E_c^{(n)} &= \{(u, v) \mid u \in V_c^{(n)}, v \in V_c^{(n)}, \text{ and the } c\text{-ary representations of} \\ &\qquad \text{their labels differ by one and only one digit}\}. \end{aligned}$$

Proof of Lemma 1. The proof is by induction on n. Clearly, $C_c^{(n)}$ is K_n when $n \leq c$. Thus, $|C_c^{(n)}| = f_c(n)$ holds for all $n \leq c$. We assume that for every $k \leq n-1$,

$|C_c^{(k)}| = f_c(k)$ holds, and will prove that $|C_c^{(n)}| = f_c(n)$. For all i $(0 \le i \le c-1)$, let $V_c^{(n)}[i] = \{u \in V_c^{(n)} \mid u_1 = c - i - 1\}$, where u_1 is the LSD of the c-ary representation of u. Clearly, each $V_c^{(n)}[i]$ has $\lfloor (n+i)/c \rfloor$ nodes. Let $E_c^{(n)}[i,j]$ $(0 \le i \le j \le c-1)$ denote a set of edges in $E_c^{(n)}$ connecting nodes in $V_c^{(n)}[i]$ and $V_c^{(n)}[j]$, that is, $E_c^{(n)}[i,j] = \{(u,v) \in E_c^{(n)} \mid u \in V_c^{(n)}[i] \text{ and } v \in V_c^{(n)}[j]\}$. For each i $(0 \le i \le i-1)$, graphs $(V_c^{(n)}[i], E_c^{(n)}[i,i])$ and $C_c^{(\lfloor (n+i)/c \rfloor)} = (V_c^{(\lfloor (n+i)/c \rfloor)}, E_c^{(\lfloor (n+i)/c \rfloor)})$ are isomorphic. Thus, from the inductive assumption, we have

$$|E_c^{(n)}[i,i]| = |E_c^{(\lfloor (n+i)/c \rfloor)}| = f_c(\lfloor (n+i)/c \rfloor).$$

For all i and j $(i < j)$ no two edges in $E_c^{(n)}[i,j]$ share a node in $V_c^{(n)}[i]$, and every node in $V_c^{(n)}[i]$ is connected with an edge in $E_c^{(n)}[i,j]$. Thus, we have

$$|E_c^{(n)}[i,j]| = |V_c^{(n)}[i]| = \lfloor (n+i)/c \rfloor.$$

Therefore, we have

$$\begin{aligned}
|E_c^{(n)}| &= \sum_{i=0}^{c-1} |E_c^{(n)}[i,i]| + \sum_{0 \le i < j \le c-1} |E_c^{(n)}[i,j]| \\
&= \sum_{i=0}^{c-1} f_c(\lfloor \frac{n+i}{c} \rfloor) + \sum_{0 \le i < j \le c-1} \lfloor \frac{n+i}{c} \rfloor \\
&= \sum_{i=0}^{c-1} \{ f_c(\lfloor \frac{n+i}{c} \rfloor) + (c-i-1) \lfloor \frac{n+i}{c} \rfloor \} \\
&= f_c(n).
\end{aligned}$$

□

Proof of Lemma 2. The proof is by induction on the number of nodes in V. Clearly, no graph with n nodes has more than $n(n-1)/2$ edges. Thus, $|E| \le g_c(|V|)$ always hold when $|V| \le c$. We assume that $|E| \le g_c(|V|)$ for any subgraph $G = (V,E)$ with $|V| \le n-1$, and prove that $|E| \le g_c(|V|)$ holds for any subgraph $G = (V,E)$ with $|V| = n$. Let $G = (V,E)$ be a subgraph with n nodes. We choose any digit s and partition V into $V[0], V[1], \ldots, V[c]$ as follows:

$$V[i] = \{u \in V | u_s = i\},$$

where u_s is the s-th digit of the c-ary representation of u. We can choose s such that at least two of the s subsets are non-empty. Further, without loss of generality, we can assume that $|V[0]| \le |V[1]| \le \cdots \le |V[c-1]| < |V|$ by renumbering the indexes of $V's$. Let $E[i,j]$ $(i \le j)$ be the edges in E connecting $V[i]$ and $V[j]$, that is, $E[i,j] = \{(u,v) \in E | u \in V[i] \text{ and } v \in V[j]\}$. Since $|V[0]| \le |V[1]| \le \cdots \le |V[c-1]| < n$, we have $E[i,i] \le g_c(|V[i]|)$ $(0 \le i \le c-1)$ from the inductive assumption. Since no two edges in $E[i,j]$ share a node in V^i, $E[i,j]$ has no more than $|V[i]|$ edges. Therefore, we have

$$|E| = \sum_{i=0}^{c-1} |E[i,i]| + \sum_{0 \le i < j \le c-1} |E[i,j]|$$

Table 2: An example of matrix A.

i	n_i	q_i	r_i	0	1	2	3	4	5	6	7
0	21	2	5	2	2	2	3	3	3	3	3
1	34	4	2	4	4	4	4	4	4	5	5
2	36	4	4	4	4	4	4	5	5	5	5
3	57	7	1	7	7	7	7	7	7	7	8
4	60	7	4	7	7	7	7	8	8	8	8
5	60	7	4	7	7	7	7	8	8	8	8
6	61	7	5	7	7	7	8	8	8	8	8
7	65	8	1	8	8	8	8	8	8	8	9

$$\begin{aligned}
&\leq \sum_{i=0}^{c-1} |g_c(V[i])| + \sum_{0 \leq i < j \leq c-1} |V[i]| \\
&\leq \sum_{i=0}^{c-1} \{|g_c(V[i])| + (c-i-1)|V[i]|\} \\
&\leq g_c(|V|).
\end{aligned}$$

□

Finally, we will prove Lemma 3.

Let $n_0, n_1, \ldots, n_{c-1}$ be c integers such that $n_0 \leq n_1 \leq \cdots \leq n_{c-1}$, and let $n = n_0 + n_1 + \cdots + n_{c-1}$ be their sum. Further, let A be a $c \times c$ matrix such that each (i, j) element $A[i, j]$ is $\lfloor \frac{n_i + j}{c} \rfloor$. Clearly, the sum of each i-th row of A is

$$n_i = \sum_{j=0}^{c-1} \lfloor \frac{n_i + j}{c} \rfloor.$$

Thus, the sum of all elements in A is equal to A. Table 2 shows an example of A.

Our first task to prove Lemma 3 is to find a $c \times c$ matrix B satisfying the following two conditions:

Condition 1 each i-th row of B is a permutation of the i-th row of A, and

Condition 2 the sum of elements in the j-th column is equal to $\lfloor \frac{n+j}{c} \rfloor$, that is, $\sum_{i=0}^{c-1} B[i, j] = \lfloor \frac{n + j}{c} \rfloor$.

Let q_i and r_i be the integers satisfying $n_i = q_i c + r_i$ $(0 \leq r_i \leq c-1)$. Also, let q and r be the integers such that $n = qc + r$ $(0 \leq r \leq c-1)$. Clearly, each i-th row of A has r'_i q_i's and r_i $(q_i + 1)$'s, where $r'_i = c - r_i$. Thus, the i-th row of B should have r'_i q_i's and r_i $(q_i + 1)$'s to satisfy Condition 1. For the purpose of satisfying Condition 2, we determine each i-th row of B as follows:

Table 3: An example of matrix B.

i	n_i	q_i	r_i	r'_i	p_i	0	1	2	3	4	5	6	7
0	21	2	5	3	0	2	2	2	3	3	3	3	3
1	34	4	2	6	3	4	5	5	4	4	4	4	4
2	36	4	4	4	9	5	4	4	4	4	5	5	5
3	57	7	1	7	13	7	7	7	7	8	7	7	7
4	60	7	4	4	20	8	8	8	8	7	7	7	7
5	60	7	4	4	24	7	7	7	7	8	8	8	7
6	61	7	5	3	28	8	8	8	8	7	7	7	8
7	65	8	1	7	31	8	8	8	8	8	8	9	8
n	394	49	2			49	49	49	49	49	49	50	50

- $B[i, p_i \bmod c] = B[i, p_i + 1 \bmod c] = \cdots = B[i, p_i + r'_i - 1 \bmod c] = q_i$, and
- $B[i, p_i + r'_i \bmod c] = B[i, p_i + r_i + 1 \bmod c] = \cdots = B[i, p_i + r'_i + r_i - 1 \bmod c] = q_i + 1$,

where $p_i = r'_1 + r'_2 + \cdots + r'_{i-1}$. Table 3 shows an example of B. Clearly, each i-th row of B has r'_i q_i's and r_i $(q_i + 1)$'s, and thus, Condition 1 is satisfied. Further from the construction of B, we have, for every k $(0 \le k \le c - 1)$

- $\sum_{i=0}^{k} B[i, 0] \le \sum_{i=0}^{k} B[i, 1] \le \cdots \le \sum_{i=0}^{k} B[i, c-1]$, and
- $\sum_{i=0}^{k} B[i, c-1] - \sum_{i=0}^{k} B[i, 0] \le 1$.

This can be proved very easily by induction on k. It follows that $\sum_{i=0}^{c-1} B[i, j] = \lfloor \frac{n+j}{c} \rfloor$ holds for every j. Thus we have,

Lemma 4 *There exists a matrix B satisfying Conditions 1 and 2.*

We are now in position to prove Lemma 3 using Lemma 4.

Proof of Lemma 3. Since $f_c \le g_c$ from the definition, it suffices for the lemma to prove $f_c \ge g_c$. We prove $f_c(n) \ge g_c(n)$ for every n by induction on n. Clearly, $f_c(n) = g_c(n)$ when $n \le c$. We assume that for all $i (< n)$, $f_c(i) \ge g_c(i)$ holds, and will prove $f_c(n) \ge g_c(n)$. For any $n_0, n_1, \ldots, n_{c-1}$ such that $n_0 \le n_1 \le \ldots \le n_{c-1} < n$ and $n = n_1 + n_2 + \cdots + n_{c-1} > c$, we have,

$$\sum_{i=0}^{c-1} \{g_c(n_i) + (c - i - 1)n_i\}$$

$$
\begin{aligned}
&\le \sum_{i=0}^{c-1} f_c(n_i) + \sum_{i=0}^{c-1}(c-i-1)n_i \quad \text{(from the inductive assumption)} \\
&= \sum_{i=0}^{c-1}\sum_{j=0}^{c-1}\{f_c(\lfloor \frac{n_i+j}{c} \rfloor) + (c-j-1)\lfloor \frac{n_i+j}{c} \rfloor\} + \sum_{i=0}^{c-1}(c-i-1)n_i \\
&\quad \text{(from the definition of } f_c\text{)} \\
&= \sum_{i=0}^{c-1}\sum_{j=0}^{c-1} f_c(\lfloor \frac{n_i+j}{c} \rfloor) + (2c-2)n - \sum_{i=0}^{c-1}\sum_{j=0}^{c-1}(i+j)\lfloor \frac{n_i+j}{c} \rfloor \\
&\quad \text{(from } n_i = \sum_{j=0}^{c-1}\lfloor \frac{n_i+j}{c} \rfloor\text{)}
\end{aligned}
$$

On the other hand, we also have

$$
\begin{aligned}
f_c(n) &= \sum_{j=0}^{c-1}\{f_c(\lfloor \frac{n+j}{c} \rfloor) + (c-j-1)\lfloor \frac{n+j}{c} \rfloor\} \\
&\quad \text{(from the definition of } f_c\text{)} \\
&\ge \sum_{j=0}^{c-1} g_c(\lfloor \frac{n+j}{c} \rfloor) + \sum_{j=0}^{c-1}(c-j-1)\lfloor \frac{n+j}{c} \rfloor \\
&\quad \text{(from the inductive assumption)} \\
&\ge \sum_{i=0}^{c-1}\sum_{j=0}^{c-1}\{g_c(B[i,j])) + (c-i-1)B[i,j]\} + \sum_{j=0}^{c-1}(c-j-1)\lfloor \frac{n+j}{c} \rfloor \\
&\quad \text{(from the definition of } g_c \text{ and Lemma 4)} \\
&\ge \sum_{i=0}^{c-1}\sum_{j=0}^{c-1}\{g_c(\lfloor \frac{n_i+j}{c} \rfloor)) + (c-i-1)\lfloor \frac{n_i+j}{c} \rfloor\} + \sum_{j=0}^{c-1}(c-j-1)\lfloor \frac{n+j}{c} \rfloor \\
&\quad \text{(from Lemma 4)} \\
&= \sum_{i=0}^{c-1}\sum_{j=0}^{c-1} g_c(\lfloor \frac{n_i+j}{c} \rfloor) + (2c-2)n - \sum_{i=0}^{c-1}\sum_{j=0}^{c-1}(i+j)\lfloor \frac{n_i+j}{c} \rfloor \\
&\ge \sum_{i=0}^{c-1}\sum_{j=0}^{c-1} f_c(\lfloor \frac{n_i+j}{c} \rfloor) + (2c-2)n - \sum_{i=0}^{c-1}\sum_{j=0}^{c-1}(i+j)\lfloor \frac{n_i+j}{c} \rfloor \\
&\quad \text{(from } f_c \le g_c\text{)}
\end{aligned}
$$

Thus, we have,

$$\sum_{i=0}^{c-1}\{g_c(n_i) + (c-i-1)n_i\} \le f_c(n).$$

It follows that $g_c(n) \le f_c(n)$ holds for every $n \ge 1$. □

3. Widths and length of C_c^d

The main purpose of this section is to compute the exact values of $\mathrm{BW}(C_c^d)$, $\mathrm{CW}(C_c^d)$, and $\mathrm{TL}(C_c^d)$. More specifically, we will prove the following three theorems:

Theorem 2 *The bisection width* $\mathrm{BW}(C_c^d)$ *of* C_c^d *is*

$$\begin{cases} c^{d+1}/4 & \text{if } c \text{ is even} \\ (c+1)(c^d-1)/4 & \text{if } c \text{ is odd.} \end{cases}$$

Theorem 3 *The cut width* $\mathrm{CW}(C_c^d)$ *of* C_c^d *is*

$$\begin{cases} c(c+2)(c^d-1)/\{4(c+1)\} & \text{if } c \text{ is even and } d \text{ is even} \\ c^2\{(c+2)c^{d-1}-1\}/\{4(c+1)\} & \text{if } c \text{ is even and } d \text{ is odd} \\ (c+1)(c^d-1)/4 & \text{if } c \text{ is odd.} \end{cases}$$

Theorem 4 *The total edge length* $\mathrm{TL}(C_c^d)$ *of* C_c^d *is*

$$(c+1)c^d(c^d-1)/6.$$

Recall that $C(G,l,i) = \{(u,v) \in E \mid 0 \le l(u) < i \le l(v) \le |V|-1\}$ denote the width of a graph $G=(V,E)$ under a linear layout l at a gap i. Also, N denotes the natural order layout, i.e., $N(u) = u$ for every u. For later reference, we define $C^-(G,l,i)$ and $C^+(G,l,i)$ as follows:

$$\begin{aligned} C^-(G,l,i) &= \{(u,v) \in E \mid 0 \le l(u) \le l(v) < i\}, \\ C^+(G,l,i) &= \{(u,v) \in E \mid i \le l(u) \le l(v) \le |V|-1\}. \end{aligned}$$

In other words, $C^-(G,l,i)$ (resp. $C^+(G,l,i)$) is the set of edges connecting nodes whose positions are less than (resp. larger than or equal to) i. Clearly, a set E of edges in G is partitioned into three sets $C(G,l,i)$, $C^-(G,l,i)$, and $C^+(G,l,i)$. Thus, $|E| = |C(G,l,i)| + |C^-(G,l,i)| + |C^+(G,l,i)|$ always holds.

To prove the above theorems, we prove the following important lemma:

Lemma 5 *For any* c, d, *linear layout* l *and gap* i,

$$|C(C_c^d,l,i)| \ge |C(C_c^d,N,i)|$$

holds.

Proof. Clearly, $C^-(C_c^d,N,i)$ has exactly the same edges in $C_c^{(i)}$. Thus, we always have $|C^-(C_c^d,N,i)| = |C_c^{(i)}|$. Since the natural order layout of C_c^d is bilateral symmetry, we have $|C^+(C_c^d,N,i)| = |C^-(C_c^d,N,n-i)| = |C_c^{(n-i)}|$. Further, $C^-(C_c^d,l,i)$ and $C^+(C_c^d,l,i)$ are i-node and $(n-i)$-node subgraphs of C_c^d, respectively. Thus, from Theorem 1, $|C^-(C_c^d,l,i)| \le |C_c^{(i)}|$ and $|C^+(C_c^d,l,i)| \le |C_c^{(n-i)}|$ hold. It follows that $|C^-(C_c^d,l,i)| \le |C^-(C_c^d,N,i)|$ and $|C^+(C_c^d,l,i)| \le |C^+(C_c^d,N,i)|$. Therefore $|C(C_c^d,l,i)| \ge |C(C_c^d,N,i)|$ always holds. □

From this lemma, when computing the parameters of C_c^d, we do not have to compute the minimum over all linear layouts but only those of the natural order layout. In other words, we have

$$\begin{aligned} \mathrm{BW}(C_c^d) &= |C(C_c^d,N,\lfloor c^d/2 \rfloor)|, \\ \mathrm{CW}(C_c^d) &= \max_i |C(G,N,i)|, \\ \mathrm{TL}(C_c^d) &= \sum_{i=1}^{c^d-1} |C(G,N,i)|, \end{aligned}$$

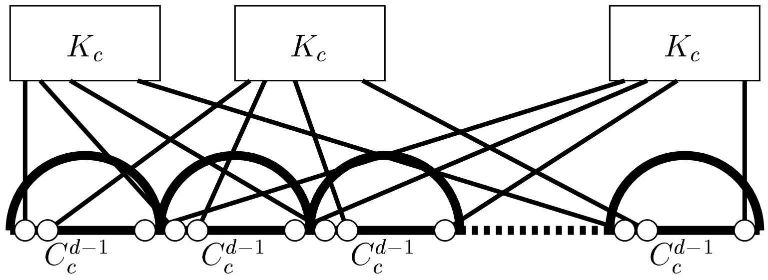

Figure 6: The natural order layout of C_c^d

for any c and d. Using this fact, we prove Theorems 2, 3, and 4.

For later reference, we define several notations. We partition all edges in C_c^d into d subsets $C_c^d[1], C_c^d[2], \ldots C_c^d[d]$ as follows:

$$C_c^d[k] \quad = \quad \{(u,v) \in C_c^d \mid u_k \neq v_k\} \qquad (1 \leq k \leq d)$$

In other words, $C_c^d[k]$ is a set of edges along the k-th dimension. Further, let $C(C_c^d[k], l, i)$ denote the edge set defined as follows:

$$C(C_c^d[k], l, i) \quad = \quad C(C_c^d, l, i) \cap C_c^d[k]$$

In other words, $C(C_c^d[k], l, i)$ is the set of edges along the k-th dimension which is separated at gap i under linear layout l.

For the reader's benefit, we will review the structure of the natural order layout of C_c^d, which is helpful to understand the forthcoming proofs. The C_c^d has c C_c^{d-1}'s as illustrated in Figure 6. Each C_c^{d-1} is also arranged by the natural order layout. Further, the corresponding nodes in c C_c^{d-1}'s are connected by K_c. More precisely, any pair of the i-th node in c C_c^{d-1}'s are connected by an edge. Thus, c C_c^{d-1}'s are connected by c^{d-1} K_c's, which correspond to $C_c^d[d]$.

Proof of Theorem 2. The bisection width of K_c is $\mathrm{BW}(K_c) = c^2/4$ if c is even, and $\mathrm{BW}(K_c) = (c^2-1)/4$ if c is odd. Thus, if c is even, $|C(C_c^d, N, c^d/2)|$ can be computed as follows:

$$\begin{aligned} |C(C_c^d, N, c^d/2)| &= c^{d-1}\mathrm{BW}(K_c) \\ &= \frac{c^{d+1}}{4}. \end{aligned}$$

If c is odd, then for all k $(1 \leq k \leq d)$,

$$\begin{aligned} |C(C_c^d[k], N, (c^d-1)/2)| &= c^{k-1}\mathrm{BW}(K_c) \\ &= \frac{(c^2-1)c^{k-1}}{4}. \end{aligned}$$

By summing up, we have

$$\begin{aligned}|C(C_c^d, N, c^d/2)| &= \sum_{k=1}^{d} |C(C_c^d[k], N, (c^d-1)/2)| \\ &= \frac{(c+1)(c^d-1)}{4}.\end{aligned}$$

□

Proof of Theorem 3. Let $i_d i_{d-1} \cdots i_1$ be the binary representation of i. From the definition of C_c^d, we have $|C(C_c^d[d], N, i)| = i(c - i_d - 1) + (c^{d-1} - i_{d-1}i_{d-2}\cdots i_1)i_d$. Hence, we have $|C(C_c^d[d], N, i+1)| - |C(C_c^d[d], N, i)| = c - 2i_d - 1$. Similarly, we can show that $|C(C_c^d[k], N, i+1)| - |C(C_c^d[k], N, i)| = c - 2i_k - 1$ for every k. It follows that the width at gap i increases by $c - 2i_d - 1$ as i increases.

First, we assume that c is odd. For every k, $|C(C_c^d[k], N, i)|$ takes the maximum when $c - 2i_k - 1 = 0$, i.e., $i_k = (c^d-1)/2$. If this is the case, $i = (c^d-1)/2$ holds, and from Theorem 2 we have,

$$\mathrm{CW}(C_c^d) = |C(C_c^d[k], N, (c^d-1)/2)| = \frac{(c+1)(c^d-1)}{4}.$$

Next, we will compute $\mathrm{CW}(C_c^d)$ when c is even. To maximize $|C(C_c^d[d], N, i)|$, we should select $i_d = c/2$ or $c/2-1$. Similarly, to maximize $|C(C_c^d[d-1], N, i)|$, we select $i_{d-1} = c/2$ or $c/2-1$. Further, we have $|C(C_c^d[d], N, i+1) \cup C(C_c^d[d-1], N, i+1)| - |C(C_c^d[d], N, i) \cup C(C_c^d[d-1], N, i)| = 2c - 2(i_d + i_{d-1}) - 2$. Thus, $|C(C_c^d[d], N, i) \cup C(C_c^d[d-1], N, i)|$ takes maximal if $2c - 2(i_k + i_{d-1}) - 2 = 0$. Consequently, we should select $i_d = c/2$ and $i_{d-1} = c/2-1$. By the same discussion, we should select $i_d = i_{d-2} = i_{d-4} = \cdots = c/2$ and $i_{d-1} = i_{d-3} = \cdots = c/2-1$ to maximize $|C(C_c^d[d-1], N, i)|$. If this is the case,

$$i = \begin{cases} \dfrac{(c+2)(c^d-1)}{2(c+1)} & \text{if } d \text{ is even,} \\ \dfrac{(c+2)c^d - c}{2(c+1)} & \text{if } d \text{ is odd.} \end{cases}$$

For such i, we have

$$|C(C_c^d[k], N, i) \cup C(C_c^d[k-1], N, i)| = \frac{(c-1)(c+2)c^{k-1}}{4}.$$

By summing up, we have

$$|C(c_c^d, N, i)| = \begin{cases} c(c+2)(c^d-1)/\{4(c+1)\} & \text{if } c \text{ is even and } d \text{ is even,} \\ c^2\{(c+2)c^{d-1}-1\}/\{4(c+1)\} & \text{if } c \text{ is even and } d \text{ is odd.} \end{cases}$$

□

Proof of Theorem 4. The total edge length of K_c is

$$\begin{aligned}\sum_{i=1}^{c-1} |C(K_n, N, i)| &= \sum_{i=1}^{c-1} i(c-i) \\ &= \frac{c(c^2-1)}{6}\end{aligned}$$

Since, $C_c^d[1]$ has c^{d-1} K_c's, we have

$$\begin{aligned}\sum_{i=1}^{c^d-1} |C(C_c^d[1], N, i)| &= c^{d-1} \cdot \frac{c^{2k-1}(c^2-1)}{6} \\ &= \frac{c^d(c^2-1)}{6}.\end{aligned}$$

In general, for all k

$$\begin{aligned}\sum_{i=1}^{c^d-1} |C(C_c^d[k], N, i)| &= c^{d-k} \cdot \frac{c^{2k-1}(c^2-1)}{6} \\ &= \frac{c^{d+k-1}(c^2-1)}{6}.\end{aligned}$$

By summing up, we have

$$\begin{aligned}\sum_{i=1}^{c^d-1} |C(C_c^d, N, i)| &= \sum_{k=1}^{d} \frac{c^{d+k-1}(c^2-1)}{6} \\ &= (c+1)c^d(c^d-1)/6.\end{aligned}$$

□

Since C_c^d has $(c-1)dc^d/2$ edges, we can compute the average edge length of C_c^d from Theorem 4.

Corollary 1 *The average edge length of C_c^d is*

$$(c+1)(c^d-1)/\{3d(c-1)\}.$$

4. Widths and length of A_c^d

The main purpose of this section is to evaluate $\mathrm{BW}(A_c^d)$, $\mathrm{CW}(A_c^d)$, and $\mathrm{TL}(A_c^d)$. More specifically, we will prove the following three theorems:

Theorem 5 *The bisection width $\mathrm{BW}(A_c^d)$ of A_c^d is*

$$\begin{cases} c^{d-1} & \text{if } c \text{ is even,} \\ (c^d-1)/(c-1) & \text{if } c \text{ is odd.} \end{cases}$$

Theorem 6 *The cut width $\mathrm{CW}(A_c^d)$ of A_c^d is at least*

$$\begin{cases} (c+2)(c^d-1)/\{c(c+1)\} & \text{if } c \text{ is even and } d \text{ is even,} \\ \{(c+2)c^{d-1}-1\}/(c+1) & \text{if } c \text{ is even and } d \text{ is odd,} \\ (c^d-1)/(c-1) & \text{if } c \text{ is odd,} \end{cases}$$

and at most

$$(c^d-1)/(c-1) \qquad \text{if } c \geq 3.$$

Theorem 7 *The total edge length* $\mathrm{TL}(A_c^d)$ *of* A_c^d *is at least*

$$\begin{cases} 2(c+1)c^{d-2}(c^d-1)/3 & \textit{if } c \textit{ is even,} \\ 2c^d(c^d-1)/\{3(c-1)\} & \textit{if } c \textit{ is odd.} \end{cases}$$

and at most

$$c^{d-1}(c^d-1).$$

Recall that we have shown that $C_c^{(n)}$ is a maximum subgraph of $C_c^{(m)}$ whenever $n \leq m$ and have computed the exact values of $\mathrm{BW}(C_c^d)$, $\mathrm{CW}(C_c^d)$, and $\mathrm{TL}(C_c^d)$ using this fact. The reader may think that $A_c^{(n)}$ is also a maximum subgraph of $A_c^{(m)}$ whenever $n \leq m$, and the exact values of $\mathrm{BW}(A_c^d)$, $\mathrm{CW}(A_c^d)$, and $\mathrm{TL}(A_c^d)$ can be computed using this fact. Unfortunately, $A_c^{(n)}$ is not always a maximum subgraph. For example, $A_4^{(4)}$ is not a maximum subgraph $A_4^{(8)}$. Graph $A_4^{(4)}$ is a 4-node 3-edge subgraph of $A_4^{(8)}$, while the maximum 4-node subgraph has four edges. Hence, we use a different technique to evaluate $\mathrm{BW}(A_c^d)$, $\mathrm{CW}(A_c^d)$, and $\mathrm{TL}(A_c^d)$.

We first compute the lower bounds of $\mathrm{BW}(A_c^d)$, $\mathrm{CW}(A_c^d)$, and $\mathrm{TL}(A_c^d)$. For this purpose, we use a method similar to embedding a directed clique in A_c^d [15, pp.223].

As a preliminary, we embed K_c in L_c. Each node K_c is assigned to a node in L_c. Each edge K_c is embedded in edges of L_c as a path. It is easy to see that no edge in L_c contains more than $h(c)$ paths, where $h(c)$ is a function such that

$$h(c) \quad = \quad \begin{cases} c^2/4 & \text{(if } c \text{ is even),} \\ (c^2-1)/4 & \text{(if } c \text{ is odd).} \end{cases}$$

Next, we embedded C_c^d in A_c^d similarly. Each node u $(0 \leq u \leq c^d - 1)$ of C_c^d is assigned to node u of A_c^d. Note that each side of C_c^d is K_c while that of A_c^d is L_c. Thus, we can embedded each edge of C_c^d in edges of A_c^d as a path such that no edge in A_c^d contains more than $h(c)$ paths. Thus we have the following Lemma:

Lemma 6 *For every gap* i $(1 \leq i \leq c-1)$*,*

$$|C(A_c^d, l, i)| \quad \geq \quad \frac{|C(C_c^d, l, i)|}{h(c)}$$

Using this lemma, we have,

Corollary 2 *For every* c *and* d*,*

$$\mathrm{BW}(A_c^d) \quad \geq \quad \frac{\mathrm{BW}(C_c^d)}{h(c)},$$

$$\mathrm{CW}(A_c^d) \quad \geq \quad \frac{\mathrm{CW}(C_c^d)}{h(c)},$$

$$\mathrm{TL}(A_c^d) \quad \geq \quad \frac{\mathrm{TL}(C_c^d)}{h(c)}$$

hold.

The lower bounds in Theorems 5, 6, and 7 can be computed using this corollary combined with Theorems 2, 3, and 4 combined.

Next, we will prove the upper bounds in Theorems 5, 6, and 7. Clearly, the bisection width, the cut width, and the total edge length for the natural order layout N give the upper bounds. In other words, we have the following lemma:

Lemma 7 *For every c and d,*

$$\begin{aligned}\mathrm{BW}(A_c^d) &\le |C(A_c^d, N, \lfloor c^d/2 \rfloor|,\\ \mathrm{CW}(A_c^d) &\le \max_i |C(A_c^d, N, i)|,\\ \mathrm{TL}(A_c^d) &\le \sum_{i=1}^{|V|-1} |C(A_c^d, N, i)|,\end{aligned}$$

hold.

Using this lemma, we compute the upper bounds.

For later reference, we define a number of notations. The edge set A_c^d is partitioned into d sets, $A_c^d[1], A_c^d[2], \ldots A_c^d[d]$ as follows:

$$A_c^d[k] = \{(u,v) \in A_c^d \mid u_k \ne v_k\} \qquad (1 \le k \le d).$$

Further, let $C(A_c^d[k], l, i)$ denote the edge set defined as follows:

$$C(A_c^d[k], l, i) = C(A_c^d, l, i) \cap A_c^d[k].$$

Proof of the upper bound in Theorem 5. If c is even, $|C(A_c^d, N, c^d/2)|$ can be computed as follows:

$$\begin{aligned}|C(A_c^d, N, c^d/2)| &= |C(A_c^d[d], N, c^d/2)|\\ &= c^{d-1}.\end{aligned}$$

If c is odd, then for all k,

$$|C(A_c^d[k], N, c^d/2)| = c^{k-1}.$$

Therefore, by summing up, we obtain

$$\begin{aligned}|C(A_c^d, N, (c^d-1)/2)| &= \sum_{k=1}^{d} |C(A_c^d[k], N, c^d/2)|\\ &= (c^d-1)/(c-1).\end{aligned}$$

□

Proof of the upper bound in Theorem 6. Let $i_d i_{d-1} \cdots i_1$ be the binary representation of i. From the definition of A_c^d, we have, for every k and i,

$$|C(A_c^d[k], N, i)| \quad \begin{cases} = c^{k-1} & \text{if } 1 \le i_k \le c-2,\\ \le c^{k-1} & \text{if } i_k = 0 \text{ or } c-1.\end{cases}$$

We thus select $i_d = i_{d-1} = \cdots = i_1 = 1$ for the maximum of $|C(A_c^d, N, i)|$. For such i,

$$|C(A_c^d[k], N, i)| = c^{k-1}$$

holds. By summing up, we obtain

$$|C(A_c^d, N, i)| = (c^d-1)/(c-1).$$

□

Proof of the upper bound in Theorem 4. The total edge length of L_c is

$$\sum_{i=1}^{c-1} |C(L_c, N, i)| = c - 1.$$

Using this fact, we have, for all k

$$\sum_{i=1}^{c^d-1} |C(A_c^d[k], N, i)| = (c-1)c^{d+k-1}$$

By summing up, we have

$$\begin{aligned} \sum_{i=1}^{c^d-1} |C(A_c^d, N, i)| &= \sum_{k=1}^{d} (c-1)c^{d+k-1} \\ &= c^{d-1}(c^d - 1). \end{aligned}$$

□

5. Concluding remarks

We have presented the exact or nearly exact values of the bisection width, the cut width, and the total edge length of generalized hypercubes. Lemma 5 implies that the natural order layout of a d-dimensional c-ary clique is the optimal layout in the sense that the width of the natural order layout at each gap is smaller than or equal to that of any other layout at the same gap. Similarly to the A_c^d case, this result makes it easy to prove that the upper and lower bounds of the widths and the total length of A_c^d with wraparound edges (referred to as a d-dimensional c-ary torus [15]) are twice as large as those of A_c^d.

After the preliminary version of this paper appeared [19], several related results have been shown. For a non-decreasing sequence positive integers $k_1, k_2, \ldots, k_d$, let $C_{k_1,k_2,\ldots,k_d} = K_{k_1} \times K_{k_2} \times \cdots \times K_{k_d}$ denotes *a d-dimensional Hamming graph*[4]. Since $C_c^d = C_{c,c,\ldots,c}$, a d-dimensional Hamming graph is a generalization of a d-dimensional clique. Similarly, $A_{k_1,k_2,\ldots,k_d} = L_{k_1} \times L_{k_2} \times \cdots \times L_{k_d}$ denotes a generalization of a d-dimensional array. Azizoğlu and Eğecioğlu showed the tight value of the bisection width of $A_{k_1,k_2,\ldots,k_d}$. More specifically, they proved that $\mathrm{BW}(A_{k_1,k_2,\ldots,k_d}) = K_e + K_{e+1} + \ldots + K_d$, where e is the largest index for which k_e is even and $K_i = k_{i-1}k_{i-2}\cdots k_1$ for $2 \le i \le d$ with $K_1 = 1$. They also showed the exact value of the isoperimetric number of A_c^d defined as follows:

$$\mathrm{IN}(A_c^d) = \min_{1 \le i \le \frac{c^d}{2}} \min_{l} \frac{|C(A_c^d, l, i)|}{i}$$

They have proved that $\mathrm{IN}(A_c^d) = \frac{2}{c}$ if c is even and $\mathrm{IN}(A_c^d) = \frac{2}{c-1}$ if c is odd [3]. Further, they showed that $\mathrm{IN}(A_{k_1,k_2,\ldots,k_d}) = 1/\lfloor \frac{k_d}{2} \rfloor$ [4].

Bezrukov et al. [23] considered the problem of embedding binary hypercubes into a rectangular grid. They showed the exact solution of the congestion (i.e. the cut width for embedding on a rectangular grid) of binary hypercubes [6]. Vrt'o proved that the cut width of *the mesh of d-ary trees* of depth n is $\theta(d^{n+1})$. The readers should refer to [11] for a survey of published results for graph layouts.

The exact values of the cut width and total edge length of A_c^d still remain to be solved.

Acknowledgments

The author would like to thank Professor Ömer Eğecioğlu and Professor Imrich Vrt'o for their valuable comments.

References

1. S. G. Akl. *Parallel Sorting Algorithms.* Academic Press, 1985.
2. S. G. Akl. *The Design and Analysis of Parallel Algorithms.* Prentice Hall, 1989.
3. M. C. Azizoğlu and Ömer Eğecioğlu. The isoperimetric number of d-dimensional k-ary arrays, *International Journal of Foundations of Computer Science*, 10(3):289–300, 1999.
4. M. C. Azizoğlu and Ömer Eğecioğlu. The bisection width and the isoperimetric number of arrays, to appear in *Discrete Applied Mathematics*,
5. B. Becker and H. U. Simon. How robust is the n-Cube ? In *Proc. of 27th Symposium on Foundation of Computer Science*, pages 283–291, October 1986.
6. S. L. Bezrukov, J. D. Chavez, L. H. Harper, M. Röttger, U.-P. Schroeder. The congestion of n-Cube layout on a rectangular grid, *Discrete Mathematics*, 213:13–19, 2000.
7. L. N. Bhuyan and D. P. Agrawal. Generalized hypercube and hyperbus structures for a computer network. *IEEE Transactions on Computers*, C-33(4), April 1984.
8. G. Brebner. Relating routing and two-dimensional grids. In P. Bertolazii and F. Luccio, editors, *VLSI: Algorithms and Architectures*, pages 221–231. Elsevier Science Publishers B.V.(North-Holland), 1985.
9. W. J. Dally. Performance analysis of k-ary n-cube interconnection networks. *IEEE Transactions on Computers*, 39(6):775–785, June 1990.
10. R. A. DeMillo, S. C. Eisenstat, and R. J. Lipton. Preserving average proximity in arrays. *Communications of the ACM*, 21(3):228–231, March 1978.
11. J. Diaz, J. Petit, and M. Serna. A survey on graph layout problems to appear in *ACM Computing Surveys*, 2002.
12. M. R. Garey, D. S. Johnson, and L. Stockmeyer. Some simplified polynomial complete problems. *SIGACT*, pages 47–63, 1974.
13. L. H. Harper. Optimal assignments of numbers to vertices. *J. Soc. Indust. Appl. Math*, 12(1):131–135, March 1964.
14. F. T. Leighton. *Complexity Issues in VLSI: Optimal Layouts for the Shuffle-Exchange Graph and Other Networks.* MIT Press, 1983.
15. F. T. Leighton. *Introduction to Parallel Algorithms and Architectures: Arrays · Trees · Hypercubes.* Morgan Kaufmann, 1992.
16. F. Makedon and I. H. Subdorough. On minimizing width in linear layouts. *Discrete*

Applied Mathematics, 23:243–265, 1989.

17. Y. Manabe, K. Hagihara, and N.Tokura. The minimum bisection widths of the cube-connected-cycles graph and cube graph. *Trans. IEICE(D) Japan*, J76–D(6):647–654, June 1984. in Japanese.
18. K. Nakano, W. Chen, T. Masuzawa, K. Hagihara, and N. Tokura. Cut width and bisection width of hypercube graph. *IEICE Transactions*, J73-A(4):856–862, April 1990. in Japanese.
19. K. Nakano, Linear layouts of generalized hypercubes, In *Proc. of Graph-Theoretic Concepts in Computer Science (LNCS 790)*, pages 364–375, June 1993.
20. L. Niepel and P. Tomasta. Elevation of a graph. *Czechoslovak Mathematical Journal*, 31(106):475–483, 1981.
21. A. L. Rosenberg. Preserving proximity in arrays. *SIAM J. Comput.*, 4(4):443–460, December 1975.
22. C. D. Thompson. Area-time complexity for VLSI. In *Proc. of 11th Symposium on Theory of Computing*, pages 81–88. ACM, 1979.
23. I. Vrt'o. Cutwidth of the mesh of d-ary Trees. In *Proc. of European Conference on Parallel Processing*, pages 242–245, 1997.
24. K. Wada and K. Kawaguchi. Optimal bounds of the crossing number and the bisection width for generalized hypercube graphs. In *Proc. of 16th Biennial Symposium on Communications*, pages 323–326, May 1992.
25. K. Wada, H. Suzuki, and K. Kawaguchi. The crossing number of hypercube graphs. In *Proc. of 43rd Convention of IPS Japan*, pages 1–95, 1991. in Japanese.
26. S. M. Yuan. Topological properties of supercube. *Information Processing Letters*, 37:241–245, March 1991.