

**Variable-Dilation Embeddings of Hypercubes into Star Graphs:
Performance Metrics, Mapping Functions, and Routing**

Marcelo Moraes de Azevedo and Nader Bagherzadeh

Department of Electrical and Computer Engineering
University of California, Irvine – Irvine, CA 92717-2625

Shahram Latifi

Department of Electrical and Computer Engineering
University of Nevada, Las Vegas – Las Vegas, NV 89154-4026

Technical Report ECE 96-05-01 – May 1996

Variable-Dilation Embeddings of Hypercubes into Star Graphs: Performance Metrics, Mapping Functions, and Routing*

Marcelo Moraes de Azevedo[†], Nader Bagherzadeh[‡], and Shahram Latifi[‡]

[†]Dept. of Electrical and Computer Engineering [‡]Dept. of Electrical and Computer Engineering
University of California, Irvine University of Nevada, Las Vegas
Irvine, CA 92717-2625 Las Vegas, NV 89154-4026
{mazevedo, nader}@ece.uci.edu latifi@jb.ee.unlv.edu
Tel: (714) 824-8720 FAX: (714) 824-2321 Tel: (702) 895-4016 FAX: (702) 895-4075

Abstract — We discuss the problem of embedding a k -dimensional hypercube $Q(k)$ into an n -dimensional star graph $S(n)$ with load 1. Communication along the i^{th} dimension of $Q(k)$ uses paths of length at most d_i in $S(n)$, where d_i is referred to as the dilation along dimension i of $Q(k)$. We embed $Q(k)$ into $S(n)$ with variable-dilation, i.e. d_i varies with i rather than being a constant. Our embeddings are an attractive alternative to previously known techniques, producing small average dilation (i.e., the average length of a path in $S(n)$ corresponding to a link of $Q(k)$) without sacrificing expansion. Using a custom simulation tool, we were able to characterize the congestion and average congestion of our embeddings under several combinations of mapping functions and routing algorithms in the star graph. To the best of our knowledge, these are the first embeddings of $Q(k)$ into $S(n)$ for which congestion results are known. Furthermore, our simulation results indicate that a small average congestion is induced on the links of $S(n)$.

Key words — Graph embeddings, hypercubes, interconnection networks, routing, star graphs.

1 Introduction

The star graph [1] is regarded as an attractive interconnection network for parallel processing, featuring smaller degree and diameter than a hypercube [2] of comparable size. However, the earlier introduction of hypercube networks, along with their interesting properties, has led to the development of a number of hypercube-configured parallel computers [2], and of a rich library of hypercube-compatible algorithms [3]. Despite the fact that some parallel algorithms have also been specifically devised for the star graph (e.g., sorting [4], FFT [5]), we believe that the repertory of star graphs algorithms can be significantly increased via hypercube embeddings.

Embedding hypercubes into star graphs, however, is known to be a challenging task. Due to topological differences between the two networks (e.g., degree and minimum cycle length), it is difficult to obtain an embedding that simultaneously achieves small *dilation* and *expansion*¹. The trade-off between dilation and expansion in embeddings of hypercubes into star graphs was first identified in the pioneer work by Nigam, Sahni, and Krishnamurthy [6].

*This research is supported in part by Conselho Nacional de Desenvolvimento Científico e Tecnológico (CNPq - Brazil), under the grant No. 200392/92-1.

¹Definitions for terms related to embeddings are given in Sec. 2.

In this paper, we propose an interesting solution to the problem, which is based on an embedding technique referred to as *packing* [7]. To achieve an embedding of a k -dimensional hypercube $Q(k)$ into an n -dimensional star graph $S(n)$, $k \geq (n - 1)$, we proceed as follows. We first embed a disjoint union $U = \bigcup_{j=0}^{2^{k-n+1}-1} Q_j(n-1)$, which contains 2^{k-n+1} many copies of $Q(n-1)$, into $S(n)$. These embedded copies are then hierarchically joined in $S(n)$ to form $Q(k)$. As shown later, this approach produces embeddings which achieve both small *average dilation*¹ and expansion. Other advantages which stem from our techniques include the capability of using $S(n)$ to host additional hypercube embeddings (see [7] for more on this topic). Because all of our embeddings and packings have load 1 (i.e., a single node of $S(n)$ receives at most one mapping of a hypercube node), these extra hypercube embeddings employ nodes that are not used by the original embedding of $Q(k)$ into $S(n)$.

Two other important metrics for characterizing an embedding are referred to as *congestion* and *average congestion*¹. To the best of our knowledge, none of the previous works on embeddings of hypercubes into star graphs has investigated these properties. In part, this can be attributed to the difficulty in obtaining analytical results for these two measures. We addressed this problem by developing a custom simulation tool for our embeddings, which is capable of evaluating exact measures for congestion, average congestion, and average dilation. These and other performance metrics are computed according to a number of user-selected options, which include choices of *node mapping functions*¹ for the embedding, and routing algorithms in the star graph. The paper illustrates some of the measures we obtained, pointing out the most promising combinations of mapping functions and routing algorithms. Our simulation results indicate that our variable-dilation embeddings have the added benefit of producing a small average congestion. Also included in the paper are analytical results that characterize our embeddings from the viewpoint of expansion, dilation, and *dilation along each of the hypercube dimensions*¹. Analytical upper bounds on average dilation are also given in this paper.

This paper is organized as follows. Sec. 2 defines performance metrics and the terminology used in the paper. Sec. 3 introduces some background information. Sec. 4 presents our variable-dilation embeddings and their corresponding analytical properties. Sec. 5 presents additional performance measures obtained via simulation. Sec. 6 concludes the paper.

2 Performance Metrics: Definitions and Terminology

Let $G(k)$ be a k -dimensional graph with hierarchical structure, such that $G(k+1)$ is obtained recursively from $c(k)$ many copies of $G(k)$. Several graphs belonging to the class of *Cayley graphs* have this recursive decomposition property, such as the hypercube and the star graph [1, 2]. The links connecting the $c(k)$ copies of $G(k)$ that exist within $G(k+1)$ are referred to as *dimension $(k+1)$ links*.

We denote the set of nodes and the set of links of $G(k)$ by $V(G(k))$ and $E(G(k))$, respectively. An *embedding* of $G(k)$ into $H(n)$, which we denote by $f : G(k) \mapsto H(n)$, is a mapping of $V(G(k))$ into $V(H(n))$ and of $E(G(k))$ into paths of $H(n)$. $G(k)$ and $H(n)$ are respectively referred to as

the *guest* and the *host* of f [3]. The *node image* of f is $f(V(G(k))) = \{f(u) : u \in V(G(k))\}$. The *load* of f is the maximum number of nodes of $G(k)$ that are mapped to any single node of $H(n)$, and is denoted by $\lambda(f)$. The *congestion* of f , denoted by $cg(f)$, is the maximum number of times any link of $H(n)$ is used by paths corresponding to the mappings of the links of $G(k)$. The *dilation* of f is $d(f) = \max\{dist_H(f(u), f(v)) : (u, v) \in E(G(k))\}$, where $dist_H(a, b)$ is the distance in $H(n)$ between two vertices a and b of $H(n)$. The *expansion* of f is $X(f) = |V(H(n))|/|V(G(k))|$.

Load, congestion, dilation, and expansion are often used to measure the quality of an embedding, and ideally should be kept as small as possible. Further characterization of an embedding, however, can be achieved by means of a few additional performance metrics, which are defined in the remainder of this section.

Let $E_i(G(k))$ denote the subset of dimension i links in $E(G(k))$, and let $f : G(k) \mapsto H(n)$ be an embedding of $G(k)$ into $H(n)$. The *dilation of f along the i^{th} dimension of $G(k)$* is $d_i(f) = \max\{dist_H(f(u), f(v)) : (u, v) \in E_i(G(k))\}$. Hence, $d(f) = \max\{d_i(f) : 1 \leq i \leq k\}$. f is referred to as a *variable-dilation* embedding if $d_i(f) < d(f)$, for at least one dimension i of $G(k)$, $1 \leq i \leq k$. Accordingly, f is referred to as a *fixed-dilation* embedding if $d_i(f) = d(f)$, $\forall i$, $1 \leq i \leq k$. The *dilation vector* of f is $\overline{d}(f) = [d_1(f), d_2(f), \dots, d_k(f)]$. The *average dilation* of f is:

$$d_{avr}(f) = \frac{\sum_{(u,v) \in E(G(k))} dist_H(f(u), f(v))}{|E(G(k))|} \quad (1)$$

An upper bound for the average dilation of an embedding f is:

$$d_{avr}(f) \leq \frac{\sum_{i=1}^k |E_i(G(k))| \cdot d_i(f)}{|E(G(k))|} \quad (2)$$

It is often the case that some dimension i links in $E_i(G(k))$ are mapped into paths of length less than $d_i(f)$ (both in variable- and fixed-dilation embeddings). Thus, Equation 1 should be used for an exact evaluation of the average dilation of an embedding.

A major advantage of variable-dilation embeddings, as opposed to fixed-dilation embeddings, is that they often produce a significantly smaller average dilation. This measure is regarded as a good approximation for the communication slowdown induced by an embedding, and has been used as a standard performance metric in practical evaluations of embedding heuristics [8].

We now define some metrics that offer a more insightful characterization of the congestion produced by an embedding $f : G(k) \mapsto H(n)$. In the following discussion, we assume that f maps the links of $G(k)$ into simple paths in $H(n)$. Let $f(u, v)$ denote the path in $H(n)$ into which link (u, v) of $G(k)$ is mapped, and let (x, y) be a link of $H(n)$. The congestion induced by (u, v) into (x, y) , denoted by $cg_{(x,y)}(f(u, v))$, is 1 if $f(u, v)$ traverses (x, y) , and 0 otherwise. The congestion induced by f into (x, y) is $cg_{(x,y)}(f) = \sum_{(u,v) \in E(G(k))} cg_{(x,y)}(f(u, v))$. Thus, $cg(f) = \max\{cg_{(x,y)}(f) : (x, y) \in E(H(n))\}$. The *link image* of f is $f(E(G(k))) = \{(x, y) : cg_{(x,y)}(f) \geq 1, (x, y) \in E(H(n))\}$. The *average congestion* of f , denoted by $cg_{avr}(f)$, is:

$$cg_{avr}(f) = \frac{\sum_{(x,y) \in f(E(G(k)))} cg_{(x,y)}(f)}{|f(E(G(k)))|} \quad (3)$$

Note that only those links of $H(n)$ that are actually used by f are included in the computation of the average congestion, thereby providing a more realistic metric. The average congestion is an important performance metric of an embedding and, along with the average dilation, gives indication on the induced communication slowdown.

Also of interest are metrics aimed at algorithms which employ only a fraction of the links of the guest graph at any point of their execution. For example, the congestion produced by algorithms based on the SIMD model of computation can be captured by the following metrics. Assume a step of a SIMD algorithm in which only dimension i links of $G(k)$ are used. The *congestion induced by dimension i links* of $G(k)$ into $H(n)$ is $cg(f(E_i(G(k)))) = \max\{\sum_{(u,v) \in E_i(G(k))} cg_{(x,y)}(f(u,v)) : (x,y) \in E(H(n))\}$. Similar metrics can be defined for algorithms that use at most $2, 3, \dots, k$ dimensions of $G(k)$ at any time.

We also note that an embedding $f : G(k) \mapsto H(n)$ can be uniquely specified by a *node mapping function* $f_V : V(G(k)) \mapsto V(H(n))$ and a deterministic routing algorithm r_H of $H(n)$. Thus, a link (u, v) of $G(k)$ is mapped to a path $r_H(f_V(u), f_V(v))$. The characterization of variable-dilation embeddings of hypercubes into star graphs provided in this paper is extensive, and includes all of the metrics defined above. Moreover, our results were obtained over a selection of four different node mapping functions and four different routing algorithms.

3 Background

3.1 The hypercube

A k -dimensional hypercube graph $Q(k) = \{V(Q(k)), E(Q(k))\}$ contains 2^k nodes, which are labeled with binary strings of length k . A node $\phi = q_1q_2 \dots q_i \dots q_k$ is connected to k distinct nodes, respectively labeled with strings $\phi_i = q_1q_2 \dots \bar{q}_i \dots q_k$, $1 \leq i \leq k$, where \bar{q}_i denotes the binary negation of bit q_i [2]. The link connecting ϕ and ϕ_i is a *dimension i link* of $Q(k)$.

3.2 The star graph

An n -dimensional star graph $S(n) = \{V(S(n)), E(S(n))\}$ contains $n!$ nodes which are labeled with the $n!$ possible permutations of n distinct symbols. In this paper, we use the integers $\{1, 2, \dots, n\}$ to label the nodes of $S(n)$. A node $\pi = p_1p_2 \dots p_i \dots p_n$ is connected to $(n - 1)$ distinct nodes, respectively labeled with permutations $\pi_i = p_i p_2 \dots p_{i-1} p_1 p_{i+1} \dots p_n$, $2 \leq i \leq n$ (i.e. π_i 's label is obtained by exchanging the first and the i^{th} symbol of π 's label) [1]. The link connecting π and π_i is a *dimension i link* of $S(n)$. Figure 1 shows $S(4)$.

3.3 Node mapping functions

Our embeddings of $Q(k)$ into $S(n)$ use two-step node mapping functions. Initially, we employ a node mapping function $h_V : V(Q(k)) \mapsto V(M(n - 1))$, where $M(n - 1)$ denotes an $(n - 1)$ -dimensional mesh of size $2 \times 3 \times \dots \times n$. The second step uses a node mapping function $g_V : V(M(n - 1)) \mapsto V(S(n))$. Thus, the composite node mapping function $f_V : V(Q(k)) \mapsto V(S(n))$ of an embedding

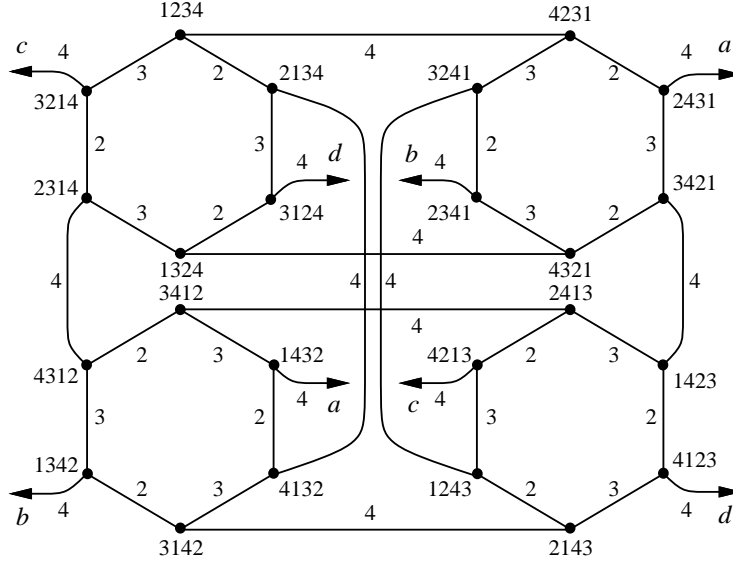


Figure 1: A 4-dimensional star graph ($S(4)$)

$f : Q(k) \mapsto S(n)$ maps node $u \in V(Q(k))$ to node $f_V(u) = g_V(h_V(u)) \in V(S(n))$. We use the operator \odot to denote the composition of node mapping functions, such that $f_V = h_V \odot g_V$.

Using $M(n-1)$ as an intermediary reference network for our embeddings provides a convenient way of dealing with the topological differences between $Q(k)$ and $S(n)$. It is known that $M(n-1)$ can be embedded into $S(n)$ with load 1, expansion 1, and dilation 3 [9, 10]. In [7], we introduced node mapping functions that support multiple node-disjoint embeddings of hypercubes into $M(n-1)$.

In the remainder of this subsection, we describe four different node mapping functions $g_V : V(M(n-1)) \mapsto V(S(n))$, which we denote by g_V^{nonh} , g_V^{mnonh} , g_V^{hier} , and g_V^{qhier} . g_V^{nonh} is referred to as the *non-hierarchical mapping function*, and was independently proposed by Jwo et al. [9] and by Ranka et al. [10]. g_V^{mnonh} is referred to as the *modified non-hierarchical mapping function*, and was introduced in [7]. g_V^{hier} and g_V^{qhier} are respectively referred to as the *hierarchical mapping function* and the *quasi-hierarchical mapping function*, and are introduced in this paper. A node mapping function $h_V : V(Q(k)) \mapsto V(M(n-1))$, which is common to all of our embeddings, is presented in Section 4. The four selections of g_V studied in this paper produce four different composite node mapping functions $f_V : V(Q(k)) \mapsto V(S(n))$, which are respectively denoted by $f_V^{nonh} = h_V \odot g_V^{nonh}$, $f_V^{mnonh} = h_V \odot g_V^{mnonh}$, $f_V^{hier} = h_V \odot g_V^{hier}$, and $f_V^{qhier} = h_V \odot g_V^{qhier}$.

We label the nodes of $M(n-1)$ with an $(n-1)$ -integer vector $m_1 m_2 \dots m_{n-1}$, where $0 \leq m_i \leq i$. A node mapping function $g_V : V(M(n-1)) \mapsto V(S(n))$ maps $\omega \in V(M(n-1))$ to $\pi \in V(S(n))$, where $\pi = g_V(\omega)$. g_V can be described algorithmically as follows. Let $p_1 p_2 \dots p_n$ be a permutation of n symbols. We denote the transposition of *symbols* i and j in $p_1 p_2 \dots p_n$ by $(i j)_s$. We define an operator \circ which applies transpositions to permutations, such that $p_1 \dots p_{k-1} p_k p_{k+1} \dots p_{\ell-1} p_\ell p_{\ell+1} \dots p_n \circ (i j)_s = p_1 \dots p_{k-1} p_\ell p_{k+1} \dots p_{\ell-1} p_k p_{\ell+1} \dots p_n$, if either $p_k = i$ and $p_\ell = j$, or $p_k = j$ and $p_\ell = i$. Similarly, we denote the transposition of the symbols occupying the i^{th} and the j^{th} positions in $p_1 p_2 \dots p_n$ by $(i j)_p$, i.e. $p_1 \dots p_{i-1} p_i p_{i+1} \dots p_{j-1} p_j p_{j+1} \dots p_n \circ (i j)_p = p_1 \dots p_{i-1} p_j p_{i+1} \dots p_{j-1} p_i p_{j+1} \dots p_n$. As an example, $2413 \circ (2 4)_s = 4213$, and $2413 \circ (2 4)_p = 2314$.

Algorithm 1 gives a general description for a node mapping function $g_V : V(M(n-1)) \mapsto V(S(n))$, which applies to g_V^{nonh} , g_V^{mnonh} , g_V^{hier} , and g_V^{qhier} . $choose(func_type, m_1 m_2)$ selects an initial permutation π_0 from Table 1. The choice for π_0 depends on the specific type of mapping function g_V and the two least significant mesh coordinates of ω (i.e., $m_1 m_2$). Thereafter, the algorithm examines m_3 through m_{n-1} and applies sequences of transpositions to π_0 as follows. For each i such that $3 \leq i \leq (n-1)$, only the first m_i transpositions specified for dimension i in Table 2 are used. For example, g_V^{mnonh} maps node 102 of $M(3)$ into node $2134 \circ (4\ 3)_p \circ (3\ 2)_p = 2143 \circ (3\ 2)_p = 2413$ of $S(4)$. Table 3 depicts complete mappings of $M(3)$ into $S(4)$ produced by each of the mapping functions g_V^{nonh} , g_V^{mnonh} , g_V^{hier} , and g_V^{qhier} .

Algorithm 1 (Node mapping function $g_V : V(M(n-1)) \mapsto V(S(n))$):

```

mesh_to_star (int n, int  $\omega[ ]$ , int  $\pi[ ]$ , char func_type)
{ int i, int j, int  $\pi_0[ ]$ ;
   $\pi_0[ ] = choose(func\_type, m_1 m_2)$ ;
  for ( $i = 3$ ;  $i < n$ ;  $i = i + 1$ )
    for ( $j = 1$ ;  $j \leq m_i$ ;  $j = j + 1$ )
       $\pi_0[ ] = \pi_0[ ] \circ transpose(func\_type, i, j)$ ;
   $\pi[ ] = \pi_0[ ]$ ; }

```

$m_1 m_2$ (mesh coordinates)	Node mapping function			
	g_V^{nonh}	g_V^{mnonh}	g_V^{hier}	g_V^{qhier}
00	12345 ... n	12345 ... n	12345 ... n	12345 ... n
10	12435 ... n	21345 ... n	21345 ... n	21345 ... n
01	13245 ... n	13245 ... n	31245 ... n	23145 ... n
11	13425 ... n	23145 ... n	13245 ... n	13245 ... n
02	14235 ... n	31245 ... n	23145 ... n	32145 ... n
12	14325 ... n	32145 ... n	32145 ... n	31245 ... n

Table 1: Choices for initial permutation π_0 used by Algorithm 1

Mesh dimension (i)	Node mapping function		
	g_V^{nonh}	g_V^{mnonh}	g_V^{hier}, g_V^{qhier}
3	$(1\ 2)_s \circ (2\ 3)_s \circ (3\ 4)_s$	$(4\ 3)_p \circ (3\ 2)_p \circ (2\ 1)_p$	$(4\ 3)_s \circ (3\ 2)_s \circ (2\ 1)_s$
4	$(1\ 2)_s \circ \dots \circ (4\ 5)_s$	$(5\ 4)_p \circ \dots \circ (2\ 1)_p$	$(5\ 4)_s \circ \dots \circ (2\ 1)_s$
\vdots	\vdots	\vdots	\vdots
$n-1$	$(1\ 2)_s \circ \dots \circ (n-1\ n)_s$	$(n\ n-1)_p \circ \dots \circ (2\ 1)_p$	$(n\ n-1)_s \circ \dots \circ (2\ 1)_s$

Table 2: Sequences of transpositions used by Algorithm 1

<i>Mesh node</i>	<i>Star graph node</i>			
	g_V^{nonh}	g_V^{mnonh}	g_V^{hier}	g_V^{qhier}
000	1234	1234	1234	1234
100	1243	2134	2134	2134
010	1324	1324	3124	2314
110	1342	2314	1324	1324
020	1423	3124	2314	3214
120	1432	3214	3214	3124
001	2134	1243	1243	1243
101	2143	2143	2143	2143
011	2314	1342	4123	2413
111	2341	2341	1423	1423
021	2413	3142	2413	4213
121	2431	3241	4213	4123
002	3124	1423	1342	1342
102	3142	2413	3142	3142
012	3214	1432	4132	3412
112	3241	2431	1432	1432
022	3412	3412	3412	4312
122	3421	3421	4312	4132
003	4123	4123	2341	2341
103	4132	4213	3241	3241
013	4213	4132	4231	3421
113	4231	4231	2431	2431
023	4312	4312	3421	4321
123	4321	4321	4321	4231

Table 3: Mappings of $M(3)$ into $S(4)$ produced by g_V^{nonh} , g_V^{mnonh} , g_V^{hier} , and g_V^{qhier}

Let g^{nonh} , g^{mnonh} , g^{hier} , and g^{qhier} denote embeddings of $M(n-1)$ into $S(n)$, which are respectively produced by the node mapping functions g_V^{nonh} , g_V^{mnonh} , g_V^{hier} , and g_V^{qhier} . Table 4 depicts a few properties of these embeddings. Proofs for g^{nonh} and g^{mnonh} are given in [10] and [7], respectively. The properties of g^{hier} and g^{qhier} can be derived similarly.

Property	Embedding			
	g^{nonh}	g^{mnonh}	g^{hier}	g^{qhier}
Dilation vector	$\underbrace{[3, \dots, 3, 1]}_{n-2}$	$\underbrace{[3, \dots, 3]}_{n-1}$	$\underbrace{[1, 2, 3, \dots, 3]}_{n-3}$	$\underbrace{[3, 2, 3, \dots, 3]}_{n-2}$
Average dilation	$\leq 3 - \frac{2}{n-1}$	≤ 3	$\leq 3 - \frac{3}{n-1}$	$\leq 3 - \frac{1}{n-1}$
Dilation	3			
Expansion	1			
Load	1			

Table 4: Properties of embeddings of $M(n-1)$ into $S(n)$

Let $a_i \dots a_{n-1}$ be a vector of $(n-i)$ integers such that $0 \leq a_i \leq i$ and $i \geq 1$. For some fixed vector $a_i \dots a_{n-1}$, we denote the induced submesh formed by all nodes $m_1 \dots m_i \dots m_{n-1} \in V(M(n-1))$, such that $m_i \dots m_{n-1} = a_i \dots a_{n-1}$, by $M(n-1)[m_i \dots m_{n-1} = a_i \dots a_{n-1}]$. g_V^{hier} maps the nodes in $M(n-1)[m_i \dots m_{n-1} = a_i \dots a_{n-1}]$ to an i -dimensional substar of $S(n)$. This can be verified by noting that the nodes of $M(n-1)[m_i \dots m_{n-1} = a_i \dots a_{n-1}]$ are mapped to $i!$ distinct permutations whose last $(n-i)$ symbols are fixed. The same property is verified for g_V^{qhier} when $i \geq 3$. For this reason, g_V^{hier} and g_V^{qhier} are regarded to as *hierarchical* mapping functions. Because g_V^{nonh} and g_V^{mnonh} do not possess this property, they are regarded to as *non-hierarchical* mapping functions. g_V^{hier} and g_V^{qhier} were devised with the intent of producing both small *average dilation* and small *average congestion*, which is confirmed by our experimental results (see Sec. 5). We later show, however, that g_V^{mnonh} also produces small congestion metrics when combined with a particular routing algorithm in $S(n)$.

4 Variable-Dilation Embeddings

In this section, we introduce a node mapping function $h_V : V(Q(k)) \mapsto V(M(n-1))$. h_V can be combined with any of the mapping functions $g_V : V(M(n-1)) \mapsto V(S(n))$ given in Subsec. 3.3, producing a composite node mapping function $f_V = h_V \odot g_V$. The corresponding embedding $f : Q(k) \mapsto S(n)$ is shown to have variable dilation. Analytical expressions for the expansion $X(f)$, dilation $d(f)$, and dilation vector $\overline{d}(f)$ are provided, as well as an upper bound on the average dilation $d_{avr}(f)$.

Fig. 2 depicts an example of the node mapping function h_V we will be describing in this section. h_V produces a variable-dilation embedding $h : Q(4) \mapsto M(3)$ with $X(h) = 1.5$, $\lambda(h) = 1$, $d(h) = 2$, $\overline{d}(h) = [1, 1, 1, 2]$, and $d_{avr}(h) = 1.25$. Assume we use $f_V^{hier} = h_V \odot g_V^{hier}$ to construct a variable-dilation embedding $f^{hier} : V(Q(4)) \mapsto V(M(3))$. Using results that are presented later in this

section, it is possible to verify that $X(f^{hier}) = 1.5$, $\lambda(f^{hier}) = 1$, $d(f^{hier}) = 4$, $\overline{d}(f^{hier}) = [1, 2, 3, 4]$, and $d_{avr}(f^{hier}) \leq 2.5$.

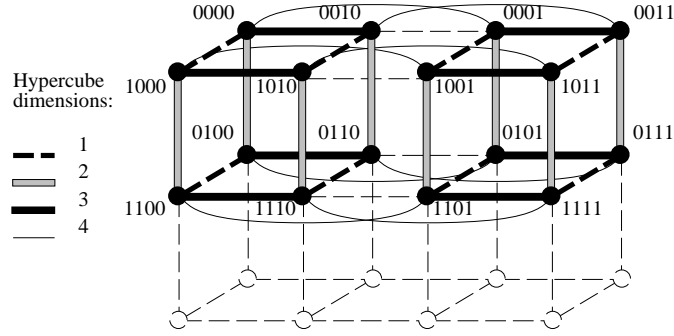


Figure 2: A variable-dilation embedding $h : Q(4) \mapsto M(3)$

A preliminary result used by our variable-dilation embeddings follows.

Lemma 1 *$Q(k)$ can be embedded into $M(n-1)$ with load 1, dilation 1, if $k \leq n-1$.*

The proof of Lemma 1 is straightforward and is omitted here. The following algorithm implements a node mapping function $h_V : V(Q(k)) \mapsto V(M(n-1))$, which produces an embedding h with the properties stated by Lemma 1.

Algorithm 2 *(Node mapping function $h_V : V(Q(k)) \mapsto V(M(n-1))$, $k \leq n-1$):*

```

cube_to_mesh (int k, int  $\phi$ [ ], int  $\omega$ [ ], int origin[ ])
{ int i;
  for ( $i = 1$ ;  $i \leq k$ ;  $i = i + 1$ )  $\omega[i] = \phi[i] + origin[i]$ ; }

```

In Alg. 2, ϕ and ω respectively denote a node of $Q(k)$ and its image in $M(n-1)$ (i.e., $\omega = h_V(\phi)$). The argument *origin* is a vector of $(n-1)$ integers, such that $origin_i < i$, $1 \leq i \leq n-1$. For $k = n-1$, the node image $h(Q(n-1))$ contains the mesh nodes that match the pattern $m_1^* \dots m_{n-1}^*$, where either $m_i^* = origin_i$ or $m_i^* = origin_i + 1$.

With multiple calls to Alg. 2 and a proper selection of the argument *origin*, one can embed multiple image-disjoint copies of $Q(n-1)$ into $M(n-1)$. This principle is the basis for *packing* hypercubes into $S(n)$ [7]. Packed $Q(n-1)$'s can be hierarchically joined to form larger hypercubes. For example, in Fig. 2, a disjoint union $Q_0(3) \cup Q_1(3)$ is packed into $M(3)$ by using calls to Alg. 2 with *origin* = 000 and *origin* = 002. The hierarchical join of $Q_0(3)$ and $Q_1(3)$ produces a variable-dilation embedding of $Q(4)$ into $M(3)$.

Theorem 1 *Let $F(x, y) = x(y+1) - 2^{x+1} + 2$, and let n , ℓ , and k be integers such that $n \geq 4$, $2 \leq \ell \leq \lfloor \log_2 n \rfloor$, and $F(\ell-1, n) < k \leq F(\ell, n)$. There is a load 1 variable-dilation embedding $h : Q(k) \mapsto M(n-1)$, whose dilation along dimension i of $Q(k)$ is:*

$$d_i(h) = \begin{cases} 1, & \text{if } 0 < i \leq F(1, n) \\ 2, & \text{if } F(1, n) < i \leq F(2, n) \\ 4, & \text{if } F(2, n) < i \leq F(3, n) \\ \vdots & \vdots \\ 2^{e-1}, & \text{if } F(e-1, n) < i \leq F(e, n), 1 \leq e < \ell \\ \vdots & \vdots \\ 2^{\ell-1}, & \text{if } F(\ell-1, n) < i \leq k, \end{cases} \quad (4)$$

Proof: Throughout the proof, we denote dimensions of $M(n-1)$ and of $Q(k)$ by \tilde{i} and i , respectively.

We hierarchically join $Q(n-1)$'s in a disjoint union $U = \bigcup_{j=0}^{2^{k-n+1}-1} Q_j(n-1)$ to form $Q(k)$. Each $Q_j(n-1)$ in U can be embedded into $M(n-1)$ with dilation 1, which is due to Lemma 1. Noting that $F(1, n) = n-1$, we have $d_i(h) = 1$, if $0 < i \leq F(1, n)$.

Let $s_i = \tilde{i} + 1$ denote the width of $M(n-1)$ along dimension \tilde{i} . If $n \geq 4$, there are $n-3$ dimensions \tilde{i} of $M(n-1)$ for which $s_i \geq 4$. Noting that $F(2, n) = 2n-4 = F(1, n) + n-3$, let $F(1, n) < i \leq F(2, n)$. $Q(i)$ can be embedded into $M(n-1)$ as follows. Let $vec[] = u_1 \dots u_z$ be a vector of z integers. Let p and s be indices such that $1 \leq p, s \leq z$, and let a be an integer. We define an operator \otimes such that $vec[] \otimes (a, p, s) = u'_p \dots u'_q \dots u'_s$, where $u'_q = a$ if $u_q \neq 0$, and $u'_q = 0$ otherwise. Thus, $\otimes(a, p, s)$ removes integers $u_p \dots u_q \dots u_s$ from $vec[]$, and replaces each non-null u_q with a . Let $bin_j[] = u_1 u_2 \dots u_{k-n+1}$ be the binary code for the index j of each $Q_j(n-1)$ in U , where u_1 is the LSB. For $0 \leq j < 2^{i-n+1}$, we embed $Q_j(n-1)$ into $M(n-1)$ using Alg. 2 with $origin(j) = \underbrace{00 \dots 0}_{2^{n-2-i}} \parallel (bin_j[] \otimes (2, 1, i-n+1))$, where \parallel denotes concatenation.

The 2^{i-n+1} copies of $Q(n-1)$ can be joined hierarchically to produce a variable-dilation embedding $h : Q(i) \mapsto M(n-1)$. Dimension i links of $Q(i)$, $F(1, n) < i \leq F(2, n)$, are mapped to paths which contain two dimension \tilde{i} links of $M(n-1)$, $\tilde{i} = i - n + 3$. Thus, $d_i(h) = 2$, if $F(1, n) < i \leq F(2, n)$. For example, a variable-dilation embedding $h : Q(6) \mapsto M(4)$ is formed by augmenting the packing $P : Q_0(4) \cup Q_1(4) \cup Q_2(4) \cup Q_3(4) \mapsto M(4)$ with the dimension 5 and 6 links that exist in $Q(6)$. In this example, $origin(0) = 0000$, $origin(1) = 0020$, $origin(2) = 0002$, and $origin(3) = 0022$.

Extensions of the technique described above can be derived for $n \geq 8$, $n \geq 16$, and so on. In general, for each $e \geq 1$, if $n \geq 2^e$, there are $n - 2^e + 1$ dimensions \tilde{i} of $M(n-1)$ for which $s_i \geq 2^e$. One can form $n - 2^e + 1$ distinct dimensions i of $Q(k)$ as follows. Dimension i links of $Q(k)$, such that $\sum_{e'=1}^{e-1} (n - 2^{e'} + 1) \leq i \leq \sum_{e'=1}^e (n - 2^{e'} + 1)$ (or $F(e-1, n) \leq \tilde{i} \leq F(e, n)$), are mapped into paths containing 2^{e-1} dimension $(i - F(e-1, n) + 2^e - 2)$ links of $M(n-1)$, which produces dilation $d_i(h) = 2^{e-1}$. This holds if $e < \ell$, where ℓ is selected such that $F(\ell-1, n) \leq k \leq F(\ell, n)$, under the restriction $2 \leq \ell \leq \lfloor \log_2 n \rfloor$. \square

Algorithm 3 implements a node mapping function $h_V : V(Q(k)) \mapsto V(M(n-1))$ according to the technique described in the proof of Theorem 1.

Algorithm 3 (Node mapping function $h_V : V(Q(k)) \mapsto V(M(n-1))$):

```

var_cube_to_mesh (int n, int k, int  $\phi$ [ ], int  $\omega$ [ ])
  { int i;
  for (i = 1; i < n; i++)  $\omega[i] = 0$ ;
  for (e = 1;  $F(e-1, n) < k$ ; e = e + 1)
    for (i =  $F(e-1, n) + 1$ ; i  $\leq$  min( $F(e, n), k$ ); i = i + 1)
       $\omega[i - F(e-1, n) + 2^e - 2] = \omega[i - F(e-1, n) + 2^e - 2] + 2^{e-1}\phi[i]$ ; }

int F(int x, int y)
  { return( $x(y+1) - 2^{x+1} + 2$ ); }

```

The composite node mapping functions f_V^{nonh} , f_V^{mnonh} , f_V^{hier} , and f_V^{qhier} can be generically described by the following algorithm:

Algorithm 4 (Node mapping function $f_V : V(Q(k)) \mapsto V(S(n))$):

```

var_cube_to_star (int n, int k, int  $\phi$ [ ], int  $\pi$ [ ], char funct_type)
  { int i, int  $\omega$ [ ];
  var_cube_to_mesh (n, k,  $\phi$ ,  $\omega$ );
  mesh_to_star (n,  $\omega$ ,  $\pi$ , funct_type) }

```

Lemma 2 gives some interesting properties of the mapping functions g_V discussed in Subsection 3.3. These properties relate to the dilation in $S(n)$ produced along dimension i links of $Q(k)$, when $i > n - 1$. For example, it is known that dimension n links of $Q(k)$ have dilation $d_n(h) = 2$ in $M(n-1)$ (Theorem 1). Moreover, $h : Q(k) \mapsto M(n-1)$ maps dimension n links of $Q(k)$ to a path containing exactly two dimension 3 links of $M(n-1)$. Using this fact and Table 4, one should expect that dimension n links of $Q(k)$ have dilation $d_n(f) = 2 \times 3$ in $S(n)$. In fact, an embedding $f : Q(k) \mapsto S(n)$ using any of the composite node mapping functions $f_V = h_v \odot g_V$ yields $d_n(f) = 4$, which is due to Lemma 2.

Lemma 2 *Let $\omega = m_1 \dots m_{i-1} m_i m_{i+1} \dots m_{n-1}$ and $\omega_{i,\theta} = m_1 \dots m_{i-1} (m_i + \theta) m_{i+1} \dots m_{n-1}$ be mesh nodes connected by a path containing θ dimension i links of $M(n-1)$, where $1 \leq i < n$ and $1 \leq \theta \leq i$. Let g_V be one of the node mapping functions g_V^{nonh} , g_V^{mnonh} , g_V^{hier} , and g_V^{qhier} , and let $dist_S(g_V(\omega), g_V(\omega_{i,\theta}))$ denote the distance in $S(n)$ between the images $g_V(\omega)$ and $g_V(\omega_{i,\theta})$. Then:*

$$dist_S(g_V^{nonh}(\omega), g_V^{nonh}(\omega_{i,\theta})) \leq \begin{cases} \theta + 2, & \text{if } 0 < i < n - 1 \\ \theta, & \text{if } i = n - 1 \end{cases} \quad (5)$$

$$dist_S(g_V^{mnonh}(\omega), g_V^{mnonh}(\omega_{i,\theta})) \leq \theta + 2 \quad (6)$$

$$dist_S(g_V^{hier}(\omega), g_V^{hier}(\omega_{i,\theta})) \begin{cases} = 1, & \text{if } i = 1 \\ \leq \theta + 1, & \text{if } i = 2 \\ \leq \theta + 2, & \text{if } 3 \leq i \leq n - 1 \end{cases} \quad (7)$$

$$\text{dist}_S(g_V^{qhier}(\omega), g_V^{qhier}(\omega_{i,\theta})) \begin{cases} = 1, & \text{if } i = 1 \text{ and } m_2 \leq 1 \\ = 3, & \text{if } i = 1 \text{ and } m_2 = 2 \\ \leq \theta + 1, & \text{if } i = 2 \\ \leq \theta + 2, & \text{if } 3 \leq i \leq n - 1 \end{cases} \quad (8)$$

Proof: Omitted. The interested reader is referred to [7] for a proof for g_V^{mnonh} . The remaining cases can be derived similarly.

The following theorem states the properties of our variable-dilation embeddings $f : Q(k) \mapsto S(n)$.

Theorem 2 Let $F(x, y) = x(y + 1) - 2^{x+1} + 2$, and let n , ℓ , and k be integers such that $n \geq 4$, $2 \leq \ell \leq \lfloor \log_2 n \rfloor$, and $F(\ell - 1, n) < k \leq F(\ell, n)$. Let $\gamma_i = 0$ if $i = F(z, n)$, for some z such that $1 \leq z \leq \ell$, and $\gamma_i = 2$ otherwise. Let f_V be one of the node mapping functions f_V^{nonh} , f_V^{mnonh} , f_V^{hier} , and f_V^{qhier} , and let $f : Q(k) \mapsto S(n)$ be one of the corresponding embeddings f^{nonh} , f^{mnonh} , f^{hier} , and f^{qhier} generated by f_V . Then, $\lambda(f) = 1$, $X(f) = \frac{n!}{2^k}$, $d(f) = \max_i \{d_i(f)\}$, and $d_{avr}(f) \leq \frac{\sum_{i=1}^k d_i(f)}{k}$, where:

$$d_i(f^{nonh}) = \begin{cases} 1 + \gamma_i, & \text{if } 0 < i \leq F(1, n) \\ 2 + \gamma_i, & \text{if } F(1, n) < i \leq F(2, n) \\ \vdots & \vdots \\ 2^{e-1} + \gamma_i, & \text{if } F(e-1, n) < i \leq F(e, n), 1 \leq e < \ell \\ \vdots & \vdots \\ 2^{\ell-1} + \gamma_i, & \text{if } F(\ell-1, n) < i \leq k, \end{cases} \quad (9)$$

$$d_i(f^{mnonh}) = \begin{cases} 3, & \text{if } 0 < i \leq F(1, n) \\ 4, & \text{if } F(1, n) < i \leq F(2, n) \\ \vdots & \vdots \\ 2^{e-1} + 2, & \text{if } F(e-1, n) < i \leq F(e, n), 1 \leq e < \ell \\ \vdots & \vdots \\ 2^{\ell-1} + 2, & \text{if } F(\ell-1, n) < i \leq k, \end{cases} \quad (10)$$

$$d_i(f^{hier}) = d_i(f^{qhier}) = \begin{cases} 1, & \text{if } i = 1 \\ 2, & \text{if } i = 2 \\ 3, & \text{if } 2 < i \leq F(1, n) \\ 4, & \text{if } F(1, n) < i \leq F(2, n) \\ 6, & \text{if } F(2, n) < i \leq F(3, n) \\ \vdots & \vdots \\ 2^{e-1} + 2, & \text{if } F(e-1, n) < i \leq F(e, n), 1 \leq e < \ell \\ \vdots & \vdots \\ 2^{\ell-1} + 2, & \text{if } F(\ell-1, n) < i \leq k, \end{cases} \quad (11)$$

Proof: Follows from Theorem 1 and Lemma 2. \square

5 Routing and Simulation Results

5.1 Preliminaries

We developed a simulation tool to characterize other important metrics of our variable-dilation embeddings. These include exact measures for the average dilation, average congestion, congestion, and congestion induced by dimension i links of $Q(k)$ (see Section 2 for definitions). Currently, the tool supports all of the node mapping functions f_V^{nonh} , f_V^{mnonh} , f_V^{hier} , and f_V^{qhier} .

Performance metrics are produced for any combination of node mapping function f_V and a routing algorithm r_S in $S(n)$. We recall that a complete characterization of congestion metrics is only possible when the specification of an embedding $f : Q(k) \mapsto S(n)$ includes a mapping of links of $Q(k)$ into paths of $S(n)$. The previously defined node mapping functions f_V give only partial knowledge about such paths. Namely, a link $(u, v) \in E(Q(k))$ is mapped to a path $f(u, v)$ with endpoints $f_V(u)$ and $f_V(v)$. However, it is often the case that a shortest path between $f_V(u)$ and $f_V(v)$ in $S(n)$ is not unique [11].

Paths in $S(n)$ can be uniquely specified if a deterministic routing algorithm r_S is assumed for $S(n)$. While this approach does not precisely capture highly non-deterministic routing algorithms, it still provides valuable insight into routing algorithms that are only partially non-deterministic (e.g., adaptive algorithms with a dominant routing rule).

Our simulation tool currently supports four deterministic routing algorithms in $S(n)$, which are described in the next subsection.

5.2 Routing algorithms in $S(n)$

The currently available selections of routing algorithms are fairly simple and do not include heuristics to optimize congestion. Some investigation on augmenting the algorithms with congestion-optimizing heuristics is the object of ongoing research. The results obtained so far, however, do not indicate significant improvements on the congestion metrics (at least for the more favorable combinations of node mapping function f_V and routing algorithm r_S). This is due to uniformity considerations: good combinations of f_V and r_S produce a fairly uniform distribution of congestion on the links of $S(n)$, which produces small average congestion (see Subsec. 5.3).

In the remainder of this subsection, we discuss four deterministic routing algorithms, which we denote by r_S^{can} , r_S^{rcan} , r_S^{ecan} , and r_S^{ocan} . We refer to these algorithms as *canonical routing*, *reverse canonical routing*, *even-only canonical routing*, and *odd-only canonical routing*, respectively. A common feature of these algorithms is that they all compute shortest-length paths. Thus, the average dilation metric computed for an embedding does not depend on the choice of routing algorithm. Another positive aspect of the currently supported routing algorithms is that they can be easily rendered in hardware.

Let $\pi_s = f_V(u)$ and $\pi_d = f_V(v)$ respectively denote the source and the destination of a path $f(u, v)$ used by $f : Q(k) \mapsto S(n)$. Due to symmetry properties of $S(n)$, routing from π_s to π_d is equivalent to routing from π_{ds} to π_0 , where $\pi_{ds} = \pi_d^{-1} \circ \pi_s$, $\pi_0 = 123 \dots n$ is the *identity node* of $S(n)$, and π_d^{-1} is the *inverse* or *reciprocal* of permutation π_d [1].

We may organize the symbols of permutation π_{ds} as a set of *cycles* – i.e. cyclically ordered

sets of symbols with the property that each symbol's desired position is that occupied by the next symbol in the set. A permutation $\pi_{ds} = 265431$, for example, can be written in cyclic format as $(1\ 2\ 6)(3\ 5)(4)$. Note that any symbol already in its correct position appears as a 1-cycle. The cyclic representation of a permutation is not unique: the cycles can appear in any order, and the symbols within each cycle may be rotated. A standard representation is referred to as the *canonical format* [12], in which: 1) the smallest symbol appears first in each cycle, and 2) cycles are ordered from left to right in decreasing order according to their first symbol. Equivalently, the *reverse canonical format* is one in which: 1) the largest symbol appears first in each cycle which do not contain symbol 1, 2) the smallest symbol (i.e., 1) appears first in the cycle which contains symbol 1, and 3) cycles are ordered from left to right in increasing order according to their first symbol. Thus, the canonical and the reversal canonical representations of 265431 are respectively $(4)(3\ 5)(1\ 2\ 6)$ and $(1\ 2\ 6)(4)(5\ 3)$.

Let $C_a = (a_1\ a_2\ \dots\ a_r)$ be an r -cycle included in π_{ds} ($2 \leq r \leq n$). The permutation produced from π_{ds} by moving the symbols in C_a to their correct positions is $\pi_{ds} \circ (a_1\ a_2\ \dots\ a_r)_p$, where $(a_1\ a_2\ \dots\ a_r)_p$ denotes the *execution* of C_a . We can express $(a_1\ a_2\ \dots\ a_r)_p$ by a sequence of transpositions as follows [11]:

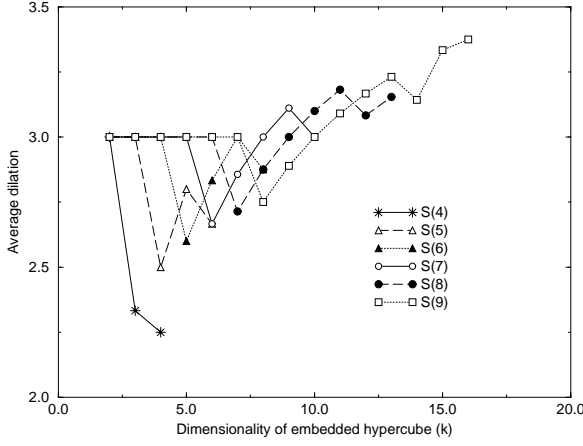
$$(a_1\ a_2\ \dots\ a_r)_p = \begin{cases} (1\ a_2)_p \circ (1\ a_3)_p \circ \dots \circ (1\ a_r)_p, & \text{if } a_1 = 1 \\ (1\ a_1)_p \circ (1\ a_2)_p \circ (1\ a_3)_p \circ \dots \circ (1\ a_r)_p \circ (1\ a_1)_p, & \text{if } a_1 \neq 1 \end{cases} \quad (12)$$

The following deterministic rules are used by the routing algorithms supported by our simulation tool. All four algorithms execute cycles from left to the right. In r_S^{can} , π_{ds} is always written in canonical format. In r_S^{rcan} , π_{ds} is always written in reverse canonical format. r_S^{ecan} uses the canonical format if the source node $\pi_s = f_V(u)$ is even-ranked, and the reverse canonical format otherwise. r_S^{ocan} is the opposite of r_S^{ecan} . A simple convention determines if the rank of $f_V(u) \in V(S(n))$ is even or odd, and consists of verifying the LSB of u 's label, $u \in V(Q(k))$.

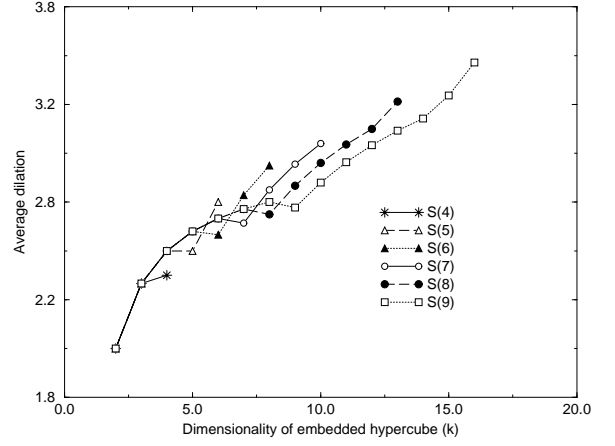
5.3 Simulation results

Figures 3 and 4 respectively depict the average dilation and the average congestion produced by our variable-dilation embeddings. Each plot in those figures corresponds to one of the node mapping functions f_V presented earlier in the paper. Moreover, in Figure 4, the routing algorithm r_S which produces the smallest average congestion for a given node mapping function is indicated. A summary of performance metrics is given in Table 5, which is intended to help the reader identify combinations of f_V and r_S which produce overall minima on a number of metrics, for several possible embeddings $f: Q(k) \mapsto S(n)$. We consider the cases $2 \leq k \leq 16$ and $4 \leq n \leq 9$, which correspond respectively to hypercubes of sizes 4 through 65,536, and star graphs of sizes 24 through 362,880.

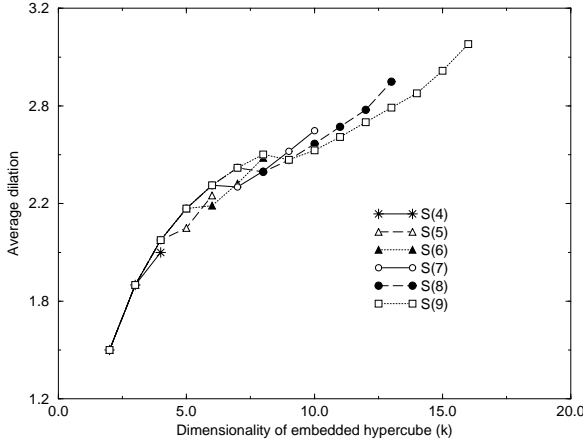
As far as average dilation is concerned, f_V^{hier} achieves the best results among all embeddings shown in Table 5. Values shown in Figures 3a, 3b, 3c, and 3d lie in the ranges $[2.25, 3.34]$, $[2.00, 3.45]$, $[1.50, 3.07]$, and $[1.50, 3.09]$, respectively. In fact, f_V^{hier} achieves average dilation which is only slightly better than that produced by f_V^{qhier} . As a rule of thumb, hierarchical mapping functions achieve smaller average dilation than the non-hierarchical mapping functions f_V^{nonh} and f_V^{mnonh} .



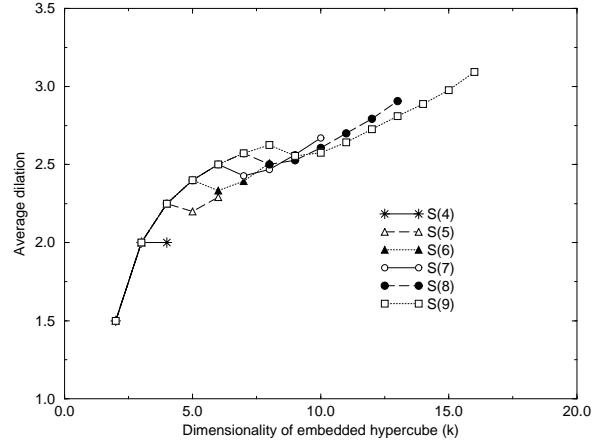
a) Node mapping function = f_V^{nonh}



b) Node mapping function = f_V^{mnonh}



c) Node mapping function = f_V^{hier}



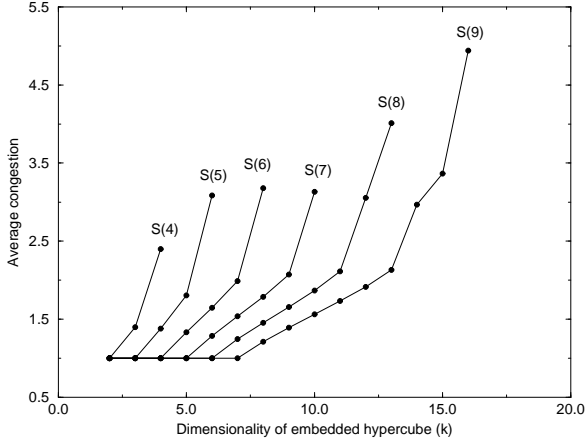
d) Node mapping function = f_V^{qhier}

Figure 3: Average dilation of embeddings of $Q(k)$ into $S(n)$

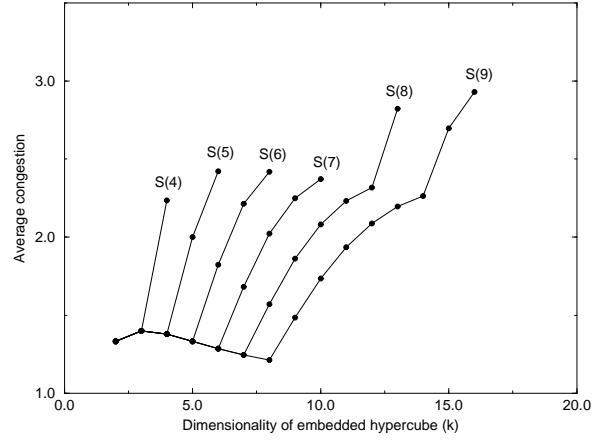
As far as routing algorithms and average congestion are concerned, we made the following observations. Non-hierarchical mapping functions perform considerably better when used in combination with the canonical routing algorithm r_S^{can} . Using either one of f_V^{nonh} or f_V^{mnonh} with r_S^{can} increases all of the congestion metrics. Hierarchical mapping functions have an opposite behavior, and should be selected in star graphs with reverse canonical routing capabilities. r_S^{ecan} and r_S^{ocan} produced congestion metrics which lie between the minima and maxima obtained with r_S^{can} (r_S^{can}) and r_S^{can} (r_S^{can}), when used in combination with hierarchical and non-hierarchical mapping functions, respectively.

f_V^{qhier} , used in combination with r_S^{can} , achieves smaller average congestion in all but one embedding shown in Table 5. The combination $f_V^{qhier} + r_S^{can}$ also demonstrates the capability of producing congestion 1 embeddings of $Q(k)$ into $S(n)$, when $k \leq n - 1$. Because f_V^{qhier} produces average dilation which is very close to that produced by f_V^{hier} , the combination $f_V^{qhier} + r_S^{can}$ appears to be a strong candidate for minimizing the communication slowdown of our embeddings.

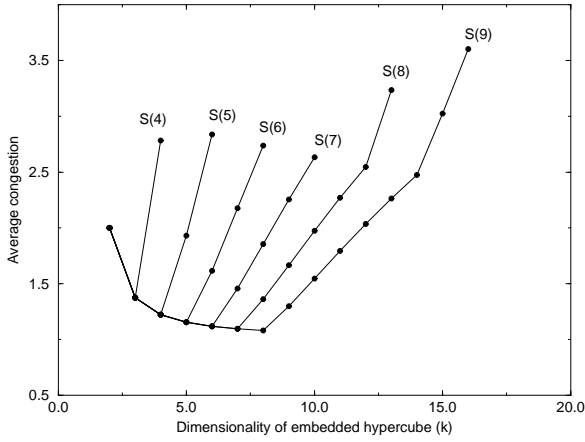
Values shown in Figures 4a, 4b, 4c, and 4d lie in the ranges $[1.00, 4.94]$, $[1.21, 2.93]$, $[1.08, 3.60]$, and $[1.00, 3.01]$, respectively. f_V^{mnonh} , used in combination with r_S^{can} , also performs quite well,



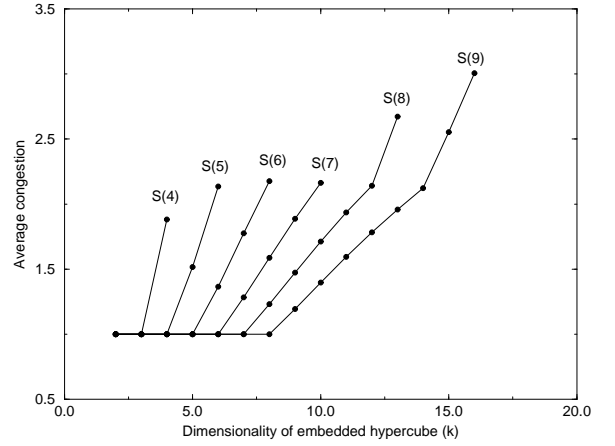
a) Node mapping function = f_V^{nonh} , canonical routing



b) Node mapping function = f_V^{mnonh} , canonical routing



c) Node mapping function = f_V^{hier} , reverse canonical routing



d) Node mapping function = f_V^{qhier} , reverse canonical routing

Figure 4: Average congestion of embeddings of $Q(k)$ into $S(n)$

especially in embeddings of large hypercubes. In fact, minima on congestion are often produced by this combination when $k \geq 10$. An analysis of log files produced by our simulator reveals that $f_V^{mnonh} + r_S^{can}$ consistently produces a fairly uniform distribution of congestion on the links of $S(n)$, which explains this result.

The careful reader might notice a few larger measures for dilation and congestion in Table 5, particularly in embeddings of large hypercubes into $S(n)$. It should be noted, however, that our embeddings are aimed at producing small *average dilation* and small *average congestion*. These metrics provide a good approximation for the communication slowdown induced by an embedding [8], and should be minimized if performance is at stake. For all the embeddings shown in Table 5, the average dilation and the average congestion will respectively lie in the ranges [1.50, 3.09] and [1.00, 3.01], if the combination $f_V^{qhier} + r_S^{can}$ is used. To further illustrate the discussion, Table 6 compares some of our variable-dilation embeddings with those of Nigam, Sahni, and Krishnamurthy

Embedding $Q(k) \mapsto S(n)$	Average dilation	Average conges- tion	Dila- tion	Con- ges- tion	Congestion due to dimension i links of $Q(k)$		
					$i \leq k - 2$	$i = k - 1$	$i = k$
$Q(2) \mapsto S(n), n \geq 4$	1.50 (c*,d*)	1.00 (a,d)	2 (c*,d*)	1 (a,d)	-	≤ 2 (all)	≤ 2 (all)
$Q(3) \mapsto S(4)$	1.83 (c*)	1.00 (d)	3 (all*)	1 (d)	≤ 2 (all)	≤ 2 (all)	≤ 2 (all)
$Q(3) \mapsto S(n), n \geq 5$	1.83 (c*)	1.00 (a,d)	3 (all*)	1 (a,d)			
$Q(4) \mapsto S(4)$	2.00 (c*)	1.88 (d)	3 (a*)	4 (c,d)			
$Q(4) \mapsto S(5)$	2.06 (c*)	1.00 (d)	3 (all*)	1 (d)	≤ 2 (all)	≤ 2 (all)	≤ 2 (all)
$Q(4) \mapsto S(n), n \geq 6$	2.06 (c*)	1.00 (a,d)	3 (all*)	1 (a,d)			
$Q(5) \mapsto S(5)$	2.12 (c*)	1.52 (d)	4 (all*)	4 (c,d)			
$Q(5) \mapsto S(6)$	2.23 (c*)	1.00 (d)	3 (all*)	1 (d)	≤ 2 (all)	≤ 2 (all)	≤ 2 (all)
$Q(5) \mapsto S(n), n \geq 7$	2.22 (c*)	1.00 (a,d)	3 (all*)	1 (a,d)			
$Q(6) \mapsto S(5)$	2.29 (c*)	2.14 (d)	4 (all*)	5 (a,d)			
$Q(6) \mapsto S(6)$	2.24 (c*)	1.37 (d)	4 (all*)	4 (c,d)	≤ 2 (all)	≤ 2 (all)	≤ 2 (all)
$Q(6) \mapsto S(7)$	2.34 (c*)	1.00 (d)	3 (all*)	1 (d)			
$Q(6) \mapsto S(n), n \geq 8$	2.34 (c*)	1.00 (a,d)	3 (all*)	1 (a,d)			
$Q(7) \mapsto S(6)$	2.35 (c*)	1.77 (d)	4 (all*)	5 (a,d)			
$Q(7) \mapsto S(7)$	2.33 (c*)	1.28 (d)	4 (all*)	4 (c,d)	≤ 2 (all)	≤ 2 (all)	≤ 2 (all)
$Q(7) \mapsto S(8)$	2.43 (c*)	1.00 (d)	3 (all*)	1 (d)			
$Q(7) \mapsto S(n), n \geq 9$	2.43 (c*)	1.00 (a,d)	3 (all*)	1 (a,d)			
$Q(8) \mapsto S(6)$	2.48 (c*)	2.18 (d)	4 (all*)	7 (d)			
$Q(8) \mapsto S(7)$	2.42 (c*)	1.59 (d)	4 (all*)	5 (a,d)	≤ 2 (all)	≤ 2 (all)	≤ 2 (all)
$Q(8) \mapsto S(8)$	2.41 (c*)	1.23 (d)	4 (all*)	4 (c,d)			
$Q(8) \mapsto S(9)$	2.50 (c*)	1.00 (d)	3 (all*)	1 (d)			
$Q(9) \mapsto S(7)$	2.52 (c*)	1.89 (d)	4 (all*)	5 (a)			
$Q(9) \mapsto S(8)$	2.47 (c*)	1.47 (d)	4 (all*)	5 (a,d)	≤ 2 (all)	≤ 2 (all)	≤ 2 (all)
$Q(9) \mapsto S(9)$	2.47 (c*)	1.19 (d)	4 (all*)	4 (c,d)			
$Q(10) \mapsto S(7)$	2.62 (c*)	2.16 (d)	4 (all*)	8 (b)			
$Q(10) \mapsto S(8)$	2.56 (c*)	1.71 (d)	4 (all*)	5 (a)	≤ 2 (all)	2 (all)	2 (all)
$Q(10) \mapsto S(9)$	2.52 (c*)	1.40 (d)	4 (all*)	5 (a,d)			
$Q(11) \mapsto S(8)$	2.64 (c*)	1.94 (d)	4 (all*)	5 (a)	≤ 2 (all)	2 (all)	2 (all)
$Q(11) \mapsto S(9)$	2.59 (c*)	1.60 (d)	4 (all*)	5 (a)			
$Q(12) \mapsto S(8)$	2.73 (c*)	2.14 (d)	4 (all*)	8 (b)	≤ 2 (all)	2 (all)	2 (all)
$Q(12) \mapsto S(9)$	2.67 (c*)	1.78 (d)	4 (all*)	5 (a)			
$Q(13) \mapsto S(8)$	2.87 (c*)	2.67 (d)	6 (all*)	13 (b)	≤ 4 (all)	2 (all)	≤ 6 (all)
$Q(13) \mapsto S(9)$	2.74 (c*)	1.96 (d)	4 (all*)	5 (a)	≤ 2 (all)	2 (all)	2 (all)
$Q(14) \mapsto S(9)$	2.81 (c*)	2.12 (d)	4 (all*)	8 (b)	≤ 2 (all)	2 (all)	2 (all)
$Q(15) \mapsto S(9)$	2.93 (c*)	2.55 (d)	6 (all*)	13 (b)	≤ 4 (all)	2 (all)	≤ 6 (all)
$Q(16) \mapsto S(9)$	3.07 (c*)	2.93 (b)	6 (all*)	15 (b)	≤ 4 (all)	≤ 6 (all)	≤ 6 (all)
<i>Legend:</i>							
(a): Node mapping function = f^{nonh} , routing algorithm = r_S^{can}							
(b): Node mapping function = f^{mnonh} , routing algorithm = r_S^{can}							
(c): Node mapping function = f^{hier} , routing algorithm = r_S^{can}							
(d): Node mapping function = f^{qhier} , routing algorithm = r_S^{can}							
(all): applicable to (a),(b),(c), and (d)							
(a*): Node mapping function = f^{nonh} , routing algorithm = irrelevant							
(b*): Node mapping function = f^{mnonh} , routing algorithm = irrelevant							
(c*): Node mapping function = f^{hier} , routing algorithm = irrelevant							
(d*): Node mapping function = f^{qhier} , routing algorithm = irrelevant							
(all*): applicable to (a*),(b*),(c*), and (d*)							

Table 5: Summary of performance metrics

[6]. Because previous works on embeddings of $Q(k)$ into $S(n)$ did not consider congestion metrics, we do not have here a basis for comparison. We do claim, however, that our variable-dilation embeddings consistently produce both small average congestion and small average dilation, and will also achieve small dilation and small congestion for many combinations of k and n (see Table 5).

<i>Embedding</i>	<i>Expansion</i>	<i>Dilation</i> (Nigam et al.)	<i>Dilation</i> (this paper)	<i>Average dilation</i> with $f_V = f_V^{hier}$ (this paper)
$Q(8) \mapsto S(6)$	2.81	4	4	2.48
$Q(10) \mapsto S(7)$	4.92	4	4	2.62
$Q(13) \mapsto S(8)$	4.92	4	6	2.87
$Q(16) \mapsto S(9)$	5.54	4	6	3.07

Table 6: Comparison with related work

As a final comment on the performance metrics shown in Table 5, we point out that several of our embeddings produce congestion 1 or 2 on the links of $S(n)$ when a single dimension of $Q(k)$ is used. This is particularly important for algorithms which employ only a fraction of the links of $Q(k)$ at any point of their execution (e.g., algorithms based on the SIMD model of computation).

6 Conclusion

This paper presented novel techniques for embedding a hypercube into a star graph, which is considered an attractive network for parallel processing. Our embeddings are designed for performance, and consistently produce small average dilation and small average congestion. We achieve these goals simultaneously by employing variable-dilation embeddings, and a careful selection of node mapping functions and routing algorithms. Altogether, the paper discusses four options each of node mapping functions and routing algorithms, which have fairly simple specifications. An extensive performance characterization of our embeddings is given in the paper, which includes the first results on congestion metrics for embeddings of $Q(k)$ into $S(n)$ we have knowledge of. Our techniques demonstrated the possibility of embedding large hypercubes into the star graph, with corresponding small expansion while producing both small average dilation and small average congestion. On continued research, we intend to expand our investigation on congestion metrics to a related technique introduced by the authors. Such a technique is referred to as *packing*, and can produce even smaller expansion by creating multiple node-disjoint embeddings of hypercubes into a star graph.

References

- [1] S. B. Akers, D. Harel, and B. Krishnamurthy, “The Star Graph: An Attractive Alternative to the n -Cube,” *Proc. Int’l Conf. on Parallel Processing*, 1987, pp. 393-400.
- [2] J. P. Hayes and T. Mudge, “Hypercube Supercomputers,” *Proceedings of the IEEE*, Vol. 77, No. 12, December 1989, pp. 1829-1841.

- [3] F. T. Leighton, *Introduction to Parallel Algorithms and Architectures: Arrays · Trees · Hypercubes*, Morgan Kaufmann Publishers, San Mateo, California, 1992.
- [4] A. Menn and A. K. Somani, "An Efficient Sorting Algorithm for the Star Graph Interconnection Network," *Proc. Int'l Conf. on Parallel Processing*, 1990, Vol. 3, pp. 1-8.
- [5] P. Fragopoulou and S. G. Akl, "A Parallel Algorithm for Computing Fourier Transforms on the Star Graph," *IEEE Transactions on Parallel and Distributed Systems*, Vol. 5, No. 5, May 1994, pp. 525-531.
- [6] M. Nigam, S. Sahni, and B. Krishnamurthy, "Embedding Hamiltonians and Hypercubes in Star Interconnection Networks," *Proc. Int'l Conf. Parallel Processing*, 1990, pp. 340-343.
- [7] M. M. Azevedo, N. Bagherzadeh and S. Latifi, *Space- and Time-Efficient Packings and Embeddings of Hypercubes into Star Graphs*, Department of Electrical and Computer Engineering, University of California, Irvine, Technical Report ECE 94-11-01, November 1994.
- [8] W.-K. Chen, M. F. M. Stallmann, and E. F. Gehring, "Hypercube Embedding Heuristics: an Evaluation," *International Journal of Parallel Programming*, Vol. 18, No. 6, 1989, pp. 505-549.
- [9] J. S. Jwo, S. Lakshminarayanan and S. K. Dhall, "Embedding of Cycles and Grids in Star Graphs," *Journal of Circuits, Systems, and Computers*, Vol. 1, No. 1, 1991, pp. 43-74.
- [10] S. Ranka, J.-C. Wang, and N. Yeh, "Embedding Meshes on the Star Graph," *Journal of Parallel and Distributed Computing* 19, 1993, pp. 131-135.
- [11] S. Latifi, "Parallel Dimension Permutations on Star Graph," *IFIP Transactions A: Computer Science and Technology*, 1993, A23, pp. 191-201.
- [12] D. E. Knuth, *The Art of Computer Programming, Vol. 1*, Addison-Wesley, 1968, pp. 73, pp. 176-177.