A General Method for Efficient Embeddings of Graphs into Optimal Hypercubes

Volker Heun Ernst W. Mayr

Institut für Informatik der Technischen Universität München
D-80290 München, Germany
{heun|mayr}@informatik.tu-muenchen.de
http://wwwmayr.informatik.tu-muenchen.de/

Abstract. Embeddings of several graph classes into hypercubes have been widely studied. Unfortunately, almost all investigated graph classes are regular graphs such as meshes, complete trees, pyramids. In this paper, we present a general method for one-to-one embedding irregular graphs into their optimal hypercubes based on extended-edge-bisectors of graphs. An extended-edge-bisector is an edge-bisector with the additional property that a subset of the vertices is distributed more or less evenly among the two halves of the bisected graph. The dilation and congestion of the embedding depends on the quality of the extended-edge-bisector. Moreover, if the extended bisection can be efficiently computed on the hypercube, so can the embedding.

1 Introduction

Hypercubes are a very popular model for parallel computation because of their regularity and their relatively small number of interprocessor connections. Another important property of an interconnection network is its ability to simulate efficiently the communication of parallel algorithms. Thus, it is desirable to find suitable embeddings of graphs representing the communication structure of parallel algorithms into hypercubes representing the interconnection network of a parallel computer.

Embeddings of graphs with a regular structure, like rings, (multidimensional) grids, complete trees, binomial trees, pyramids, X-trees, meshes of trees and so on, have been investigated by many researcher, see, e.g., [4, 5, 6, 7, 8, 11, 15, 20, 21, 22, 25]. Unfortunately, the communication structure of a parallel algorithm can often be very irregular.

For graphs whose structure is less regular, it is in general hard to decide whether there is a good embedding into a given host graph. Here, an embedding is considered good if it has small dilation, load, and expansion. Given a graph G and a positive integer d, it is $\mathcal{NP}$-complete to decide whether G is a subgraph of the d-dimensional hypercube, even if G is a tree [23]. Given a graph G and a positive integer d, it is also $\mathcal{NP}$-complete to decide whether G has a dilation 2 embedding into the d-dimensional hypercube [24].

For arbitrary binary trees, one-to-one embeddings into their optimal hypercubes with constant dilation and constant node-congestion have been constructed in [3]. Another one-to-one embedding of an arbitrary binary tree into its optimal hypercube with constant dilation is given in [18], but in this paper neither node-congestion nor edge-congestion is considered. The embedding given in [13] yields dilation 8 and constant node-congestion. This is the best known bound on the dilation. Furthermore, this embedding can be efficiently computed on the hypercube itself. In [10], Havel has conjectured that every binary tree has a one-to-one embedding with dilation 2 into its optimal hypercube. This conjecture is still open. A simple parity argument shows that the complete binary tree is not a subgraph of its optimal hypercube [4, 20, 25]. In [12], it was shown that balanced caterpillars with legs of unit length are subgraphs of their optimal hypercubes. This was generalized in [2]; it was shown that balanced caterpillars with the additional property that the lengths of the legs have the same parity are subgraphs of their optimal hypercubes. Embeddings of graphs with bounded treewidth into optimal hypercubes have been studied in [14]. A one-to-one embedding with dilation $3\lceil\log((d{+}1)(t{+}1))\rceil{+}8$ was obtained which can be efficiently constructed on the optimal hypercube itself.

In this paper, a general technique for embedding irregular graphs based on extended-edge-bisectors is presented. An extended-edge-bisector is an edge-bisector with the additional property that a subset of the vertices is distributed more or less evenly among the two halves of the bisected graph. Depending on the size of the extended-edge-bisector, roughly the number of vertices incident to the edges cut by the bisector, the quality of distributing marked vertices, and the maximal degree of the given graph, we compute bounds on the dilation and node-congestion for our embedding method. Moreover, if the extended bisection can be efficiently computed on the hypercube, so can the embedding.

The remainder of this paper is organized as follows. First, we recall some basic definitions and notations which we will use later. In the third section, we present the main tool for our embedding, called the (k, h, o, λ)-tree, and we discuss the quality of the embedding achieved. We define extended-edge-bisectors in the fourth section and describe how to efficiently obtain extended-edge-separators from extended-edge-bisectors. We also determine the quality of the embedding in terms of the quality of the extended-edge-bisector for a given family of graphs. In the fifth section, we present some applications of our method. Finally, we give some concluding remarks.

2 Preliminaries

An *embedding* of a graph $G{=}(V_G, E_G)$, called *guest graph*, into a graph $H{=}(V_H, E_H)$, called *host graph*, is a mapping $\phi{:}G{\rightarrow}H$ consisting of two mappings $\phi_V{:}V_G{\rightarrow}V_H$ and $\phi_E{:}E_G{\rightarrow}\mathcal{P}(H)$. Here, $\mathcal{P}(G)$ denotes the set of paths in the graph $G{=}(V, E)$. The mapping ϕ_E maps each edge $\{v, w\}{\in}E_G$ to a path $p{\in}\mathcal{P}(H)$ connecting $\phi_V(v)$ and $\phi_V(w)$. We call an embedding *one-to-one* if the mapping ϕ_V is 1-1.

The *dilation* of an edge $e{\in}E_G$ under an embedding ϕ is the length of the path $\phi_E(e)$. Here, the length of a path p is the number of its edges. The *dilation of an embedding* ϕ is the maximal dilation of an edge in G. The number of vertices of a guest graph which are mapped onto a vertex v in the host graph, is called the *load* of the vertex v. The *load of an embedding* ϕ is the maximal load of a vertex in the host graph. In this paper, unless noted otherwise, we only consider embeddings with load one. The ratio $|V_H|/|V_G|$ is called the *expansion* of the embedding ϕ. The *congestion* of an edge $e'{\in}E_H$ is the number of paths in $\{\phi_E(e) \mid e{\in}E_G\}$ that contain e'. The *edge-congestion* is the maximal congestion over all edges in H. The *congestion* of a vertex $v{\in}V_H$ is the number of paths in $\{\phi_E(e) \mid e{\in}E_G\}$ containing v. Again, the *node-congestion* is the maximal congestion over all vertices in H. In the following, we initially restrict our attention to finding a suitable mapping ϕ_V, and we will use shortest paths in the hypercube for the mapping ϕ_E. Nevertheless, it is still important to decide which paths we choose, since we are interested in obtaining an embedding with small node-congestion.

A *hypercube of dimension* d is a graph with 2^d vertices, labeled 1-1 with the strings in $\{0,1\}^d$. Two vertices are connected iff their labels differ in exactly one position. The smallest hypercube into which we can embed a given graph $G{=}(V,E)$ with load one is called its *optimal* hypercube. Thus, its dimension is $\lceil \log(|V|) \rceil$. Hence, an embedding of a graph G into its optimal hypercube has expansion less than 2.

The *level* of a vertex v in a tree is the number of vertices on the path from the root to v. Hence, the level of the root is 1. The *height* of a tree T is the maximum level of a vertex in T. A *complete* d-ary tree T of height h is a tree such that each internal vertex has exactly d children, and such that all leaves have the same level. Given a vertex v in a tree, a vertex w is called a descendant of v if it lies on a path from v to a leaf of the tree. A subtree rooted at a vertex v is the induced subgraph of all descendants of v in the tree. We call a vertex of a tree an *internal* vertex if it is not a leaf of the tree.

3 The (k,h,o,λ)-Tree

3.1 Definition of a (k,h,o,λ)-Tree

To construct our embedding, we use the data structure of a (k,h,o,λ)-tree. The (k,h,o,λ)-*tree* is a complete 2^k-ary tree of height h with integer node weights, also called the *capacities* of the nodes. The capacity of a node depends on the level of the node and additional integer parameters $o{\geq}0$ and $\lambda{\in}[0{:}k{-}1]$. We will call o the order and λ the type of a (k,h,o,λ)-tree. The capacity of a node at level ℓ of a (k,h,o,λ)-tree is defined as follows ($\delta_{i,j}$ is the Kronecker-delta):

$$c(\ell) = 2^o \left[\sum_{i=0}^{k-1} 2^i \binom{k(h-l+1)-i-1}{\lambda} + \delta_{\ell,1} \sum_{i=0}^{\lambda} \binom{kh}{i} \right],$$

In the sequel, we call vertices of a (k,h,o,λ)-tree *nodes*, and we denote the capacity of a node at level ℓ by $c(\ell)$.

Lemma 1. *The total capacity of a* (k,h,o,λ)*-tree is* 2^{kh+o}.

Proof. By definition of the capacities we get:

$$\begin{aligned}
\sum_{\ell=1}^{h} 2^{k(\ell-1)} \cdot c(\ell) &= \sum_{\ell=1}^{h} 2^{k(\ell-1)} \cdot 2^o \sum_{i=0}^{k-1} 2^i \binom{k(h-\ell+1)-i-1}{\lambda} + 2^o \sum_{i=0}^{\lambda} \binom{kh}{i} \\
&= 2^o \left[\sum_{\ell=1}^{h} \sum_{i=0}^{k-1} 2^{kh-1-(k\ell-i-1)} \binom{k\ell-i-1}{\lambda} + \sum_{i=0}^{\lambda} \binom{kh}{i} \right] \\
&= 2^o \left[\sum_{i=0}^{kh-1} 2^{kh-1-i} \binom{i}{\lambda} + \sum_{i=0}^{\lambda} \binom{kh}{i} \right] \\
&\quad \text{using } \sum_{i=\lambda}^{n-1} 2^{n-1-i} \tbinom{i}{\lambda} = \sum_{i=\lambda+1}^{n} \tbinom{n}{i} \\
&= 2^o \left[\sum_{i=\lambda+1}^{kh} \binom{kh}{i} + \sum_{i=0}^{\lambda} \binom{kh}{i} \right] \quad = \quad 2^{kh+o}
\end{aligned}$$

□

We state without proof the following inequalities for $c(\ell)$ which we will use later.

Lemma 2. *For $\lambda \leq \frac{k}{2}$ the following inequalities are valid:*

$$\text{i) } \frac{1}{2} \cdot 2^{k+o} \left(\frac{k(h-\ell)}{\lambda}\right)^{\lambda} \leq \frac{3}{4} c(\ell), \quad \text{ii) } c(\ell-1) \leq 3^{\lambda+1} c(\ell), \quad \text{iii) } 2^{o}(2^{k}-1) \leq c(\ell).$$

For $\lambda=0$ the term in parentheses on the left hand side of the first inequality is set to be 1.

3.2 Embedding the (k, h, o, λ)-Tree into the Hypercube

We now describe a mapping of a (k, h, o, λ)-tree into its optimal hypercube such that each node of the (k, h, o, λ)-tree occupies as many vertices of the hypercube as given by its capacity. Each node in a (k, h, o, λ)-tree can be represented by a string in $(\{0,1\}^k)^*$ as follows. The empty string ϵ represents the root of the (k, h, o, λ)-tree. If α represents a node v of the (k, h, o, λ)-tree, then the strings $\alpha\beta$ for $\beta \in \{0,1\}^k$ represent the 2^k children of v from left to right. Note that the string representing a node at level ℓ has length $k(\ell-1)$. For a string $v \in \{0,1\}^*$, we denote by $|v|_1$ the number of 1's in v and by $|v|$ the length of v. We define the following sets of hypercube locations, where α represents some arbitrary node in a (k, h, o, λ)-tree:

$$\begin{aligned} S_\alpha &:= \left\{\alpha\beta\gamma\delta \in \{0,1\}^{kh+o} : \beta \in (0+1)^* 1 \wedge |\beta| \leq k \wedge |\gamma|_1 = \lambda \wedge |\delta| = o\right\} \\ T &:= \left\{\gamma\delta \in \{0,1\}^{kh+o} : |\gamma|_1 \leq \lambda \wedge |\delta| = o\right\} \end{aligned}$$

We also define the set $S := \bigcup_\alpha S_\alpha$. The vertices of the given graph mapped to the node of a (k, h, o, λ)-tree represented by α, will finally be mapped to hypercube locations in the set $L_\alpha := S_\alpha$ if $\alpha \neq \epsilon$, and $L_\epsilon := S_\epsilon \cup T$ otherwise. We will now show that the capacity of a node in the (k, h, o, λ)-tree is equal to the cardinality of the set of vertices in the hypercube to which it is mapped. Let α represent a node in the (k, h, o, λ)-tree at level ℓ; therefore, $|\alpha| = k(\ell-1)$. Hence we get:

$$|T| = 2^o \sum_{i=0}^{\lambda} \binom{kh}{i}, \quad |S_\alpha| = 2^o \sum_{\substack{\beta \in (0+1)^* 1 \\ |\beta| \leq k}} \binom{kh-|\alpha|-|\beta|}{\lambda} = 2^o \sum_{i=0}^{k-1} 2^i \binom{k(h-l+1)-i-1}{\lambda}.$$

Thus, $|L_\alpha| = c(\ell)$. Furthermore, it can easily be verified that for every $\alpha \neq \alpha' \in (\{0,1\}^k)^*$ $S_\alpha \cap S_{\alpha'} = \emptyset$ and $S_\alpha \cap T = \emptyset$, and hence $L_\alpha \cap L_{\alpha'} = \emptyset$ for every $\alpha \neq \alpha' \in (\{0,1\}^k)^*$. Hence, for each string $s \in S \cup T$ there is a unique decomposition $s = \alpha\beta\gamma\delta$ as used in the definition of S_α and T. For a given hypercube location v, we will call the (k, h, o, λ)-tree node represented by α such that $v \in L_\alpha$ the *corresponding* (k, h, o, λ)-tree node.

Lemma 3. *Let v and w be representations of two nodes in a (k, h, o, λ)-tree and let $u = \mathrm{lca(v, w)}$ be their lowest common ancestor in the (k, h, o, λ)-tree. If the lengths of the paths from u to v and from u to w are at most Δ then any pair r, s of hypercube locations, where $r \in L_v$ and $s \in L_w$, differ in at most $k(\Delta+1)+o+2\lambda$ positions.*

Proof. We first consider the case where both vertices r and s belong to the set S. The diagram in Figure 1 shows the labels of the two hypercube locations r and s to which v and w are mapped. In this picture, α represents the lowest common ancestor of v and w and $\alpha\alpha'$ (resp., $\alpha\alpha''$) represents the vertex v (resp., w). Without loss of generality, we

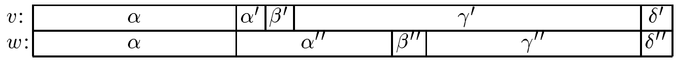

Fig. 1. Hypercube Locations of Adjacent Graph Vertices

assume $|\alpha'|{\le}|\alpha''|$. Since the lowest common ancestor of v and w is at distance $\le\Delta$ from both vertices, we get $|\alpha''|{\le}k\Delta$. The definition of the mapping from the (k,h,o,λ)-tree to the hypercube implies that $|\beta''|{\le}k$, $|\delta''|{=}o$, and that γ' and γ'' contain exactly λ 1's each. Hence, the labels r and s differ in at most $k\Delta{+}k{+}\lambda{+}\lambda{+}o{=}k(\Delta{+}1){+}o{+}2\lambda$ positions.

We now consider the case that the hypercube location r belongs to the set T, implying that v represents the root of the (k,h,o,λ)-tree. The hypercube location $s{=}\alpha'\beta'\gamma'\delta'$, belongs again to S. Since the distance between v and w is Δ, we have $|\alpha'|{\le}k\Delta$. Hence, there are at most $k\Delta{+}k{+}\lambda{=}k(\Delta{+}1){+}\lambda$ 1's in the first kh positions of s. By the definition of the set T, r has at most λ 1's in the first kh positions. Thus, the hypercube locations r and s differ in at most $k(\Delta{+}1){+}\lambda{+}\lambda{+}o{=}k(\Delta{+}1){+}o{+}2\lambda$ positions.

Finally, if both vertices r and s belong to the set T, they obviously differ in at most $o{+}2\lambda$ positions. $\square$

In the following, we will bound the node-congestion of the given embedding. We restrict our attention to bound the node-congestion, since the edge-congestion is less than or equal to the node-congestion. Moreover, for a better understandable representation we consider a $(k,h,o,1)$-tree. The argument for (k,h,o,λ)-trees is similar.

Consider two adjacent vertices of the given graph which are mapped to hypercube locations labeled v and w. We decompose the label into four segments A, B, C, and D. Segment D consists of the last o bits. The lengths of the segments A, B, and C are multiples of k; Segment C is the longest suffix before segment D such that it contains at most one 1 in both v and w and its length is a multiple of k. Segment B consists of the $(\Delta+1)k$ positions before segment C, and segment A is the remainder. See Figure 2 for an illustration of this decomposition. Recall that the hypercube locations of the vertices v and w can differ only in segments B and D, and in at most 2 positions of segment C (cf. Figure 2 for the case $v,w{\in}S$, the positions where the labels can differ are indicated by shading). Also note that segment A can be empty. For a path $p_{v,w}$ from v to w, we call v the *lower* endpoint of $p_{v,w}$, if $v{=}\alpha_v\beta_v\gamma_v\delta_v{\in}S$, $w{=}\alpha_w\beta_w\gamma_w\delta_w{\in}S$ (cf. Figure 2) and $|\gamma_v|{<}|\gamma_w|$, or if $v{\in}S$ and $w{\in}T$. Otherwise, if $v,w{\in}S$ and $|\gamma_v|{=}|\gamma_w|$ or if $v,w{\in}T$, we arbitrarily select one endpoint of the path $p_{v,w}$ to be the lower endpoint. The endpoint of the path $p_{v,w}$ which is not the lower endpoint is called the *upper* endpoint.

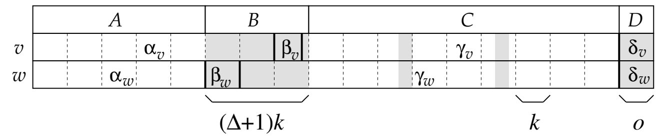

Fig. 2. Hypercube Locations of Two Adjacent Vertices

To construct a shortest path $p_{v,w}$ from v to w in the hypercube, we proceed in four phases. Without loss of generality, we assume that v is the lower endpoint; otherwise, we execute the routing about to be described in reverse order. In the first phase, we flip the bit position in segment C which has to be changed from 0 to 1, if it exists. In the second phase, we flip those bit positions in segment D that need to be changed. In the third phase, we first flip in segment B 0's to 1's from right to left that have to be changed; then, we flip 1's to 0's from left to right whenever necessary. Finally, we flip the bit position in segment C which has to be changed from 1 to 0, if it exists.

To obtain an upper bound on the node-congestion, we consider a fixed hypercube location u and bound the number of hypercube locations which can be an upper or lower endpoint of a path hitting u. The details are omitted due to space limitations.

Theorem 4. *Let G be a graph of size n and maximal degree d. Let an embedding of G into a (k, h, o, λ)-tree be given such that the load of no (k, h, o, λ)-tree node exceeds its capacity. If a pair of adjacent graph vertices is mapped to a pair of (k, h, o, λ)-tree nodes such that their lowest common ancestor is at most at distance Δ from either node, we obtain an embedding of G into a hypercube with unit load, dilation at most $(\Delta{+}1)k{+}o{+}2\lambda$, and expansion at most $2^{kh+o}/n$. Choosing the shortest paths between each pair of such hypercube locations as described above, the node-congestion of this embedding is at most $O(d2^{(\Delta+1)k+o+2\lambda})$.*

4 The Embedding

In this section, we will construct embeddings of arbitrary graphs into (k, h, o, λ)-trees provided the graphs have small bisectors. Further, if the bisectors can be efficiently constructed on the hypercube then the embedding can also be efficiently implemented on the hypercube.

4.1 Edge-Bisectors and Edge-Separators

Let $G{=}(V, E)$ be a graph containing marked vertices and let $\mu(V){\subseteq}V$ be the set of marked vertices in G. Let $\alpha, \beta{:}\mathbb{R}{\to}\mathbb{R}$ be two functions. Let $S{\subseteq}E$ be a subset of edges. We denote by $G(E')$ the subgraph $G'{=}(V, E')$ of G for some $E'{\subseteq}E$. Further, let $C_1{=}(V_1, E_1), \ldots, C_\ell{=}(V_\ell, E_\ell)$ be the connected components of the graph $G(E \backslash S)$. We call S an (α, β)*-extended-edge-bisector* if there exists a partition of the connected components of the graph $G(E \backslash S)$ into sets $\mathcal{S}_1$ and $\mathcal{S}_2$ such that the following conditions hold for $i{\in}\{1, 2\}$:

$$\textit{i)}\ \left\lfloor \frac{n}{2} \right\rfloor \le \sum_{C_j \in \mathcal{S}_i} |V_j| \le \left\lceil \frac{n}{2} \right\rceil, \quad \textit{ii)}\ |S| \le \alpha(|V|), \quad \textit{iii)}\ \sum_{C_j \in \mathcal{S}_i} |\mu(V_j)| \le \lceil \beta(|\mu(V)|) \rceil .$$

For $\beta(n){=}\frac{n}{2}$, such a separator is known in the literature as a 2-color-bisector [1, 9]. We call an edge belonging to the set S a *separator edge*. Vertices incident to a separator edge will be called *separator vertices*. We will denote the set of separator vertices in a connected component $C{=}(V, E)$ (or more generally in a graph $G{=}(V, E)$) by $\sigma(V)$.

Again, let $G{=}(V, E)$ be a graph containing marked vertices and let $\mu(V){\subseteq}V$ be the set of marked vertices in G. Let $\gamma{:}\mathbb{R}{\times}\mathbb{R}{\to}\mathbb{R}$ be a real valued function. Let $S{\subseteq}E$ be a subset of the edges. We denote by $C_1{=}(V_1, E_1), \ldots, C_\ell{=}(V_\ell, E_\ell)$ the connected components of the graph $G(E \backslash S)$. We call S a (γ, κ)*-extended-edge-separator* if there exists a partition of the connected components of the graph $G(E\setminus S)$ into sets $\mathcal{S}_i$, for $i\in[1{:}\kappa]$, such that the following conditions hold:

$$\forall i \in [1{:}\kappa]: \ \left\lfloor \frac{n}{\kappa} \right\rfloor \leq \sum_{C_j\in\mathcal{S}_i} |V_j| \leq \left\lceil \frac{n}{\kappa} \right\rceil \ \wedge \ \sum_{C_j\in\mathcal{S}_i} |\mu(V_j)\cup\sigma(V_j)| \leq \gamma(|V|,|\mu(V)|).$$

A set $\mathcal{F}$ of graphs is called a *family* of graphs if each subgraph of a graph in $\mathcal{F}$ is also in $\mathcal{F}$. More formally, for each $G{=}(V,E)\in\mathcal{F}$, for each subset $V'\subseteq V$, and for each subset $E'\subset(E\cap(V'\times V'))$, it follows that $G'{=}(V',E')\in\mathcal{F}$. A family $\mathcal{F}$ of graphs is called (α,β)-*bisectable* if each graph $G\in\mathcal{F}$ has an (α,β)-extended-edge-bisector. A family $\mathcal{F}$ of graphs is called (γ,κ)-*separable* if each graph $G\in\mathcal{F}$ has a (γ,κ)-extended-edge-separator.

4.2 Constructing Edge-Separators from Edge-Bisectors

Using a (α,β)-extended-edge-bisector, we are able to construct a $(\gamma,2^k)$-extended-edge-separator.

Theorem 5. *Let $\mathcal{F}$ be a family of (α,β)-bisectable graphs such that β is monotonically increasing, i.e., $x{\leq}y \Rightarrow \beta(x){\leq}\beta(y)$, and β satisfies the triangle inequality, i.e., $\beta(x{+}y){\leq}\beta(x){+}\beta(y)$. Then $\mathcal{F}$ is also $(\gamma,2^k)$-separable for every $k\in\mathbb{N}$, where*

$$\gamma(x,y) \ = \ \sum_{i=0}^{k-1} \left\lceil (\beta^{(i)}\circ\alpha)\left(\left\lceil \frac{x}{2^{k-1-i}} \right\rceil\right) \right\rceil + \beta^{(k)}(y) + \sum_{i=0}^{k-1} \beta^{(i)}(1).$$

If $\beta(x){=}\frac{x}{2}$ then we get for γ: $$\gamma(x,y) \ = \ \sum_{i=0}^{k-1} \left\lceil \frac{1}{2^i}\cdot\alpha\left(\left\lceil \frac{x}{2^{k-1-i}} \right\rceil\right) \right\rceil + \left\lceil \frac{y}{2^k} \right\rceil.$$

Proof. We will prove the theorem by induction on k.
Induction basis ($k{=}1$): Let $G{=}(V,E)\in\mathcal{F}$. Since $\mathcal{F}$ is a family of (α,β)-bisectable graphs, the first condition in the definition of an extended-edge-separator follows immediately from condition i) in the definition of an extended-edge-bisector. By definition, the number of marked and separator vertices in each half is at most

$$\alpha(|V|) + \lceil\beta(|\mu(V)|)\rceil \leq \alpha(|V|) + \beta(|\mu(V)|) + 1 \leq \alpha(|V|) + \beta(|\mu(V)|) + \beta^{(0)}(1),$$

as claimed in the theorem. If $\beta(x){=}\frac{x}{2}$ then the number of marked and separator vertices in each half is at most $\alpha(|V|){+}\lceil|\mu(V)|/2\rceil$.
Induction step ($k{\to}k{+}1$): First, we apply the induction hypothesis for $k{=}1$ to the graph G. Hence, we get a partition of the connected components $C_1,\ldots,C_\ell$ of the graph $G(E\setminus S)$ into two sets $\mathcal{S}_1$ and $\mathcal{S}_2$ satisfying the required conditions. Let $G^i{=}(V^i,E^i)$, for $i\in\{1,2\}$, be the union of the connected components in $\mathcal{S}_i$, i.e., $V^i{=}\bigcup_{C_j\in\mathcal{S}_i} V(C_j)$ and $E^i{=}\bigcup_{C_j\in\mathcal{S}_i} E(C_j)$. Since $\mathcal{F}$ is a family of graphs, it follows that $G^i\in\mathcal{F}$ and, therefore, the graphs G^i are (α,β)-bisectable. Now we mark all separator vertices and apply the induction hypothesis for $k{-}1$ to each graph $G^i{=}(V^i,E^i)$ in parallel. Thus, we obtain 2^k graphs $G_i{=}(V_i,E_i)$. Thus, $\lfloor\frac{n}{2^k}\rfloor{=}\lfloor\frac{\lfloor n/2\rfloor}{2^{k-1}}\rfloor{\leq}|V_i| \leq \lceil\frac{\lceil n/2\rceil}{2^{k-1}}\rceil{=}\lceil\frac{n}{2^k}\rceil$. Moreover, the number of marked and separator vertices in each G_i is $|\mu(V_i)\cup\sigma(V_i)|$

$$\le \sum_{i=0}^{k-2} \left\lceil (\beta^{(i)} \circ \alpha)\left(\left\lceil \frac{\lceil |V_i|/2 \rceil}{2^{k-2-i}} \right\rceil\right) \right\rceil + \beta^{(k-1)}\Big(\alpha(|V_i|) + \lceil \beta(|\mu(V_i)|) \rceil\Big) + \sum_{i=0}^{k-2} \beta^{(i)}(1)$$

using the triangle inequality for $\beta(\cdot)$ and the monotonicity of $\beta(\cdot)$

$$\le \sum_{i=0}^{k-1} \left\lceil (\beta^{(i)} \circ \alpha)\left(\left\lceil \frac{|V_i|}{2^{k-1-i}} \right\rceil\right) \right\rceil + \beta^{(k-1)}\big(\beta(|\mu(V_i)|) + 1\big) + \sum_{i=0}^{k-2} \beta^{(i)}(1)$$

using the triangle inequality for $\beta(\cdot)$

$$\le \sum_{i=0}^{k-1} \left\lceil (\beta^{(i)} \circ \alpha)\left(\left\lceil \frac{|V_i|}{2^{k-1-i}} \right\rceil\right) \right\rceil + \beta^{(k)}(|\mu(V_i)|) + \sum_{i=0}^{k-1} \beta^{(i)}(1) \;=\; \gamma(|V_i|, |\mu(V_i)|)$$

If $\beta(x) = \frac{x}{2}$ then the number of marked and separator vertices in each G_i is

$$\begin{aligned} |\mu(V_i) \cup \sigma(V_i)| &\le \sum_{i=0}^{k-2} \left\lceil \frac{1}{2^i} \cdot \alpha\left(\left\lceil \frac{\lceil |V_i|/2 \rceil}{2^{k-2-i}} \right\rceil\right) \right\rceil + \left\lceil \frac{\alpha(|V_i|) + \lceil |\mu(V_i)|/2 \rceil}{2^{k-1}} \right\rceil \\ &\le \sum_{i=0}^{k-1} \left\lceil \frac{1}{2^i} \cdot \alpha\left(\left\lceil \frac{|V_i|}{2^{k-1-i}} \right\rceil\right) \right\rceil + \left\lceil \frac{|\mu(V_i)|}{2^k} \right\rceil \;=\; \gamma(|V_i|, |\mu(V_i)|) \end{aligned}$$

The claim on the computation times follows immediately from the observation that the bisections in each of the k stages can be done in parallel in an optimal subcube. Further, routing of the subgraphs created by the bisection to their own subcubes can be done in logarithmic time using concentration routing [19]. $\square$

We should mention here that the hypercube algorithm for the bisection must satisfy some special requirements. Whenever the hypercube algorithm requires some conditions on how the graph is stored in the hypercube before the bisection, then the algorithm must ensure that these conditions are also true for the subgraphs resulting from the bisection.

4.3 The Embedding Algorithm

Given a family $\mathcal{F}$ of (α, β)-bisectable graphs, we now consider the embedding of a graph $G{=}(V, E){\in}\mathcal{F}$ into the hypercube. We denote the number of vertices in G by n, and the maximal degree of a vertex in G by d. In the sequel, we assume that $d{\ge}3$, since all graphs with degree ≤ 2 can easily be embedded one-to-one into its optimal hypercube.

Our embedding of (α, β)-bisectable graphs into optimal hypercubes is achieved in two steps. First, we embed the graph into a (k, h, o, λ)-tree. This will be explained in detail in the following. Then, we use the mapping presented in the previous section to complete the embedding. To obtain small dilation, adjacent vertices of the graph should be mapped to nodes which are close in the (k, h, o, λ)-tree. Our goal is to obtain an embedding of the graph into a (k, h, o, λ)-tree such that adjacent vertices are mapped to two nodes of the (k, h, o, λ)-tree with distance at most 1 from their lowest common ancestor of the (k, h, o, λ)-tree. Our method leads to an embedding of the graph G into the hypercube with dilation $2k{+}o{+}2\lambda$.

The embedding of the graph G into the (k, h, o, λ)-tree will be achieved as follows. First, we fill up the root of the (k, h, o, λ)-tree with arbitrarily chosen vertices of G and remove these vertices from G, obtaining G'. Then we mark the unmapped neighbors of the mapped vertices in the resulting graph G'. We associate this graph G' with the root of the (k, h, o, λ)-tree. We decompose G' into 2^k parts using Theorem 5 and associate to each of the children of the root one of the parts of the decomposed graph G'. In the next step, we fill up the children of the root with the marked vertices in the associated subgraph of G'. Additionally, we map the separator vertices of the previous decomposition to the 2^k children of the root. Finally, we fill up the nodes of the (k, h, o, λ)-tree with arbitrarily chosen vertices of the associated subgraph until the capacity of these nodes is reached. We repeat this process until we reach the leaves of the (k, h, o, λ)-tree.

4.4 Quality of the General Embedding Strategy

In this subsection, we will show how the parameters k, o, and λ can be chosen depending on the extended-edge-separator and the maximal degree d of the given graph to obtain an embedding with small dilation. Since we are looking for an embedding into the optimal hypercube, we have to choose the parameters of the (k, h, o, λ)-tree such that $kh+o=\lceil \log(n) \rceil$ implying that the height h is determined by k and o.

We make the following assumptions on our (α, β)-bisectable family of graphs. We recall from Theorem 5 that β is function satisfying the triangle inequality such that $\beta(x) \in [\frac{x}{2}, x]$. We first assume that $\beta{:}\mathbb{R} \to \mathbb{R}$ is a linear function. For our convenience, we will write $\beta(n)=\beta \cdot n$. Further, we assume that $\alpha{:}\mathbb{R} \to \mathbb{R}$ is a poly-logarithmic function, i.e., $\alpha(x)=\alpha \cdot \log^\lambda(x)$ for some $\alpha \in \mathbb{R}^+$.

In order to compute the number of graph vertices mapped to a single node in the (k, h, o, λ)-tree, let $n(\ell)$ be the maximum number of marked vertices and separator vertices mapped to a single node of the (k, h, o, λ)-tree at level ℓ, and let $f(\ell)$ be the size of the associated graph which is partitioned at a node at level ℓ of the (k, h, o, λ)-tree. An obvious upper bound for $f(\ell)$ is $\lceil \frac{2^{kh+o}-c(1)}{2^{k(\ell-1)}} \rceil$. Recall that $c(\ell)$ denotes the capacity of a node in the (k, h, o, λ)-tree at level ℓ. The number of marked vertices in a forest corresponding to a node at level ℓ before the partitioning is at most $(d-1)c(\ell)$, since each mapped vertex has at most $d-1$ unmapped neighbors. It what follows, we will show that $n(\ell) \leq c(\ell)$. Note that by construction of the embedding $n(1) \leq c(1)$ and $n(h) \leq c(h)$. Now we can establish the following inequality for $2 \leq \ell \leq h-1$ using Theorem 5.

$$
\begin{aligned}
n(\ell) &\leq \sum_{i=0}^{k-1} \left\lceil \beta^{(i)} \left(\alpha \left(\frac{f(\ell-1)}{2^{k-1-i}} \right) \right) \right\rceil + \beta^{(k)} \Big((d-1) \cdot c(\ell-1) \Big) + \sum_{i=0}^{k-1} \beta^{(i)} \\
&\leq \alpha \frac{1-\beta^k}{1-\beta} \log^\lambda \left(\frac{2^{kh+o}}{2^{k(\ell-2)}} \right) + k + (d-1)\beta^k c(\ell-1) + k \\
&\quad \text{using Lemma 2 } \textit{ii)} \text{ for } \lambda \leq \tfrac{k}{2} \\
&\leq \alpha \frac{1-\beta^k}{1-\beta} \left(k(h-\ell+2)+o \right)^\lambda + \beta^k (d-1) 3^{\lambda+1} c(\ell) + 2k \overset{!}{\leq} c(\ell)
\end{aligned}
$$

Thus, it is sufficient to satisfy the following three inequalities.

$$
2k \overset{!}{\leq} \frac{1}{8} c(\ell) \qquad (1)
$$

$$\beta^k 3^{\lambda+1}(d-1)c(\ell) \overset{!}{\leq} \frac{1}{8}c(\ell) \tag{2}$$

$$\alpha\frac{1-\beta^k}{1-\beta}\left(k(h-\ell+2)+o\right)^\lambda \overset{!}{\leq} \frac{1}{2}\cdot 2^{k+o}\left(\frac{k(h-\ell)}{\lambda}\right)^\lambda \leq \frac{3}{4}c(\ell) \tag{3}$$

For inequality (3) we have used Lemma 2 *i)*. Inequality (1) is satisfied if we choose $o{>}k$ and $k{\geq}3$, since using Lemma 2 *iii)*, we get:

$$2k \quad \leq \quad 2^{2k-3} \quad \leq \quad 2^{k+o-4} \quad \leq \quad \frac{1}{8}2^o(2^k-1) \quad \leq \quad \frac{1}{8}c(\ell).$$

From inequality (2) we may conclude for $\lambda{>}0$ that

$$k \quad \geq \quad \frac{\log(d)+\lambda\log(3)+4}{-\log(\beta)}.$$

Clearly, inequality (3) is satisfied if the following inequality holds, assuming that $o{\leq}2k$.

$$2\alpha\frac{1-\beta^k}{1-\beta}\lambda^\lambda\left(1+\frac{2k+o}{k(h-\ell)}\right)^\lambda \quad \leq \quad 2\alpha\frac{1}{1-\beta}2^{\lambda\log(5\lambda)} \quad \overset{!}{\leq} \quad 2^{k+o}$$

Hence, the following inequalities should be satisfied, if we choose $o{\in}[k{+}1{:}2k]$:

$$k+o \overset{!}{\geq} \lambda\log(5\lambda)+\log\left(\frac{2\alpha}{1-\beta}\right) \quad \Rightarrow \quad k \overset{!}{\geq} \frac{1}{2}\left(\lambda\log(5\lambda)+\log\left(\frac{2\alpha}{1-\beta}\right)\right)$$

Altogether, if we choose k and o as described above, we never map more vertices to a (k,h,o,λ)-tree node at level $\ell{\leq}h{-}1$ than its capacity allows. Using Lemma 3, we can conclude that the dilation of the embedding is at most $2k{+}o{+}2\lambda{\leq}4k{+}2\lambda$.

Theorem 6. *Let $\mathcal{G}$ be a family of (α,β)-bisectable graphs, where $\alpha(n){=}\alpha'\cdot\log^\lambda(n)$ is a poly-logarithmic function and $\beta(x){\leq}\beta'{\cdot}x$. Then a graph $G{\in}\mathcal{G}$ can be one-to-one embedded into its optimal hypercube with dilation at most $4k+2\lambda$ and node-congestion at most $O(d2^{4k+2\lambda})$, where*

$$k = \left\lceil \max\left\{\tfrac{1}{2}\left(\lambda\log(5\lambda)+\log\left(\tfrac{2\alpha}{1-\beta}\right)\right), -(\log(d)+\lambda\log(3)+4)/\log(\beta)\right\}\right\rceil$$

Moreover, if a (α,β)-bisection can be computed in time $\mathcal{B}(n){=}\Omega(\log(n))$ for any graph of size n on its optimal hypercube, this embedding can computed in time $O(\log(n){\cdot}\mathcal{B}(n))$ on the optimal hypercube.

Obviously, the dilation and node-congestion can be improved for special graph classes by a more sophisticated analysis of the inequality $n(\ell){\leq}c(\ell)$.

5 Applications

In this section, we will apply our results of the previous sections to some special families of graphs. Some of the results are well known, but others are new. It has been shown in [13] that a binary tree has a $(4\log(n), \frac{n}{2})$-extended-edge-bisector. As shown in [13], this extended-edge-bisector can be computed on the optimal hypercube in logarithmic time after some preprocessing requiring time $O(\log^2(n)\,\mathrm{logloglog}(n)\log^*(n))$.

Theorem 7. *An arbitrary binary tree can be embedded into its optimal hypercube with constant dilation, constant node-congestion, and unit load. The embedding can be computed in time $O(\log^2(n)\,\mathrm{logloglog}(n)\log^*(n))$ on the optimal hypercube itself.*

The graphs with treewidth t and maximal degree d have a $(2(t+1)(d+1)\log(n), \frac{n}{2})$-extended-edge-bisector, as shown in [14].

Theorem 8. *A graph with treewidth* t *and maximal degree* d *can be embedded into its optimal hypercube with dilation* $\max\{14+\lceil 2\log((d+1)(t+1))\rceil, 24+4\lceil\log(d)\rceil\}$. *The embedding can be computed in time* $O(\log^2(n)\,\mathrm{logloglog}(n)\log^*(n))$ *on the optimal hypercube itself if the input graph is given by its tree decomposition.*

It can also be shown that graphs with pathwidth t and maximal degree d have a $((t+1)(d+1), \frac{n}{2})$-extended-edge-bisector.

Theorem 9. *A graph with pathwidth* t *and maximal degree* d *can be embedded into its optimal hypercube with dilation* $\max\{4+\lceil 2\log((d+1)(t+1))\rceil, 16+4\lceil\log(d)\rceil\}$. *The embedding can be computed in time* $O(\log^2(n)\,\mathrm{logloglog}(n)\log^*(n))$ *on the optimal hypercube itself if the input graph is given by its tree decomposition.*

Using the results in [1, 9], it can be shown that the family of circular-arc graphs have a $(d(d+1), \frac{n}{2})$-extended-edge-bisector, with d the maximal degree of a vertex in the graph.

Theorem 10. *A circular-arc graph with maximal degree* d *can be embedded into its optimal hypercube with dilation* $4\lceil\log(d)\rceil+4$. *The embedding can be computed in time* $O(\log^2(n)\,\mathrm{logloglog}(n)\log^*(n))$ *on the optimal hypercube itself if the input graph is given by its circular-arc representation.*

Since interval graphs are a subclass of circular-arc graphs, it follows:

Corollary 11. *An interval graph with maximal degree* d *can be embedded into its optimal hypercube with dilation* $4\lceil\log(d)\rceil+4$. *The embedding can be computed in time* $O(\log^2(n)\,\mathrm{logloglog}(n)\log^*(n))$ *on the optimal hypercube itself if the input graph is given by its interval representation.*

The family of k-outerplanar graphs has a $(6k(d+1)\log(n), \frac{n}{2})$-extended-edge-bisector.

Theorem 12. *A* k*-outerplanar graph can be one-to-one embedded into its optimal hypercube with dilation at most* $\max\{2\lceil\log(k(d+1))\rceil+8, 26+4\lceil\log(d)\rceil\}$.

6 Conclusion

We have presented a general technique for embedding irregular graphs based on extended-edge-bisectors. Depending on the size of the extended-edge-bisector, the quality of distributing marked vertices, and the maximal degree of the given graph, we have given bounds on the dilation and node-congestion for our embedding. It remains as an open problem whether an embedding can be computed dynamically as the graph grows. It has been proven in [17] that any deterministic algorithm for dynamically embedding caterpillars into hypercubes cannot simultaneously achieve constant dilation, constant load, and constant expansion. Thus, either randomized techniques (as in [17]) or migration, i.e., remapping of vertices, has to be used for dynamic embeddings. In [16], a deterministic algorithm for dynamic one-to-one embeddings of arbitrary binary trees into hypercubes with small dilation, expansion, and node-congestionusing migration is presented. Furthermore, this embedding can be efficiently computed on the hypercube. It seems that this method can at least be extended for k-trees.

References

1. N. Alon, D. West: The Borsuk-Ulam Theorem and Bisection of Necklaces, *Proc. Am. Math. Soc.*, **98**(1986), 623–628.
2. S. Bezrukov, B. Monien, W. Unger,G. Wechsung, Embedding Ladders and Caterpillars into the Hypercube, Preprint, GH-Univ. Paderborn, 1995, (to appear in *Disc. Appl. Math.*).
3. S. Bhatt, F. Chung, T. Leighton, A. Rosenberg: Efficient Embeddings of Trees in Hypercubes, *SIAM J. Comput.*, **21**(1992), 151–162.
4. S. Bhatt, I. Ipsen: How to Embed Trees in Hypercubes, *Yale Univ.*, RR-443, 1985.
5. M. Chan: Embedding of d-Dimensional Grids into Optimal Hypercubes, *Proc. of the 1989 Symp. on Parallel Algorithms and Architectures*, 52–57.
6. M. Chan: Embedding of Grids into Optimal Hypercubes, *SIAM J. Comput*, **20**(1991), 834–864.
7. M. Chan, F. Chin, C. Chu, W. Mak: Dilation-5 Embedding of 3-Dimensional Grids into Hypercubes, *J. Parallel Distrib. Comput.*, **33**(1996), 98-106.
8. K. Efe: Embedding Mesh of Trees in the Hypercube, *J. Parallel Distrib. Comput.*, **11**(1991), 222–230.
9. C. Goldberg, D. West: Bisection of Circle Colorings, *SIAM J. Alg. Disc. Meth.*, **6**(1985), 93–106.
10. I. Havel: On Hamiltonian Circuits and Spanning Trees of Hypercubes (in Czech.), *Časopis. Pěst. Mat.*, **109**(1984), 145–152.
11. I. Havel, P. Liebl: Embedding the Polytomic Tree into the n-Cube, *Časopis. Pěst. Mat.*, **98**(1973), 307–314.
12. I. Havel, P. Liebl: One-Legged Caterpillars Span Hypercubes, *J. Graph Theory*, **10**(1986), 69–76.
13. V. Heun, E. Mayr: A New Efficient Algorithm for Embedding an Arbitrary Binary Tree into Its Optimal Hypercube, *J. Algorithms*, **20**(1996), 375–199.
14. V. Heun, E. Mayr: Embedding Graphs with Bounded Treewidth into Optimal Hypercubes, *Proc. of the 13th Symp. on Theoretical Aspects of Computer Science*, LNCS 1046, 157–168.
15. V. Heun, E. Mayr: Optimal Dynamic Edge-Disjoint Embeddings of Complete Binary Trees into Hypercubes, (to appear in *Proc. of the 4th Workshop on Parallel Systems and Algorithms*).
16. V. Heun, E. Mayr: Efficient Dynamic Embeddings of Arbitrary Binary Trees into Hypercubes, (to appear in *Proc. of the IRREGULAR'96*).
17. T. Leighton, M. Newman, A. Ranade, W. Schwabe: Dynamic Tree Embeddings in Butterflies and Hypercubes, *SIAM J. Comput.*, **21**(1992), 639–654.
18. B. Monien, H. Sudborough: Simulating Binary Trees on Hypercubes, *Proc. of the Aegean Workshop on Computing*, LNCS 319, 170–180.
19. D. Nassimi, S. Sahni: Data Broadcasting in SIMD Computers, *IEEE Trans. Comput.*, **C-30**(1981), 101-107.
20. Y. Saad, M. Schulz: Topological Properties of the Hypercube, *Yale Univ.*, RR-389, 1985.
21. X. Sheen, Q. Hu, W. Liang: Embedding k-ary Complete Trees into Hypercubes, *J. Parallel Distrib. Comput.*, **24**(1995), 100–106.
22. Q. Stout: Hypercubes and Pyramids, *Proc. of the NATO Advanced Research Workshop on Pyramidal Systems for Computer Vision 1986*, 75–89.
23. A. Wagner, D. Corneil: Embedding Tree in a Hypercube is NP-Complete, *SIAM J. Comput.*, **19**(1990), 570–590.
24. A. Wagner, D. Corneil: On the Complexity of the Embedding Problem for Hypercube Related Graphs, *Discrete Appl. Math.*, **43**(1993), 75-95.
25. A. Wu: Embedding of Tree Networks into Hypercubes, *J. Parallel Distrib. Comput.*, **2**(1985), 238-249.