Simulating Binary Trees on X-Trees

(Extended Abstract)

Burkhard Monien*
Dept. of Math. and Computer Science
University of Paderborn
4790 Paderborn, Germany

Abstract

We show how to embed an arbitrary binary tree with dilation 11 and optimal expansion into an X-tree. To our knowledge this is the first result proving that every binary tree can be simulated by a "natural" network of bounded degree with constant dilation and constant expansion. Our construction also leads to a universal graph of bounded degree for binary trees, the degree bound being at most 415.

1 Introduction

A lot of work has been done in recent years studying the properties of interconnection networks for parallel computer systems. An important feature of an interconnection network is its degree of universality, i.e. its ability to simulate programs written for other architectures without a significant time delay. The popularity of the hypercube network is based also on the fact that it can simulate common program structures like grids or trees in a very efficient way.

In this paper we are interested in the simulation of binary trees. Binary trees reflect common data structures and the type of program structure found in common divide-and-conquer algorithms. Bhatt, Chung, Leighton and Rosenberg [1] show that arbitrary binary trees can be embedded into hypercubes with constant expansion and dilation 10. In [7] Monien and Sudborough improve this result and describe an embedding with constant expansion and dilation 3. They also show that every binary tree can be embedded into its optimal hypercube (i.e. without expansion) with dilation 5.

Hypercubes have many properties distinguishing them as an excellent candidate for an interconnection network. However their vertex degree increases with the number of vertices. Cube connected cycles and butterfly networks are networks of constant degree sharing the topological properties of the hypercube, especially they have a small diameter and a very good routing behaviour. Up to now it is not totally clear up to what extent these networks also have the good universal behaviour of the hypercube. In [3] Bhatt, Chung, Hong, Leighton and Rosenberg give a negative and a positive answer. They show that grids and X-trees cannot be embedded with constant expansion and dilation into cube connected cycles and butterfly networks. The embedding of grids needs dilation $0(\log n)$ and the embedding of X-trees dilation $0(\log\log n)$, where n is the number of nodes. These are the first graphs that are known to be efficiently embeddable into hypercube networks but not into cube connected cycles or into butterfly networks. On the other hand they show that complete binary trees can be embedded with dilation $0(1)$ and expansion $0(1)$. The efficiency of simulat-

*This work was supported by the grant Mo 285/4 from the German Research Association (DFG).

ing arbitrary binary trees is left open. To our knowledge there exists no result showing that arbitrary binary trees can be embedded into some "natural" network of small degree with dilation 0(1) and expansion 0(1). The existence of such a "universal" network of bounded degree is known ([1,2,6]), but the previous constructions lead to a very large vertex degree which is left unspecified.

In this paper we study embeddings of binary trees into X-trees. An X-tree is a graph that is obtained from a complete binary tree by adding cross edges connecting the vertices of the same level. The X-tree of height 3 is shown in the figure 1 below. An embedding is a mapping of the vertices of the tree into the nodes of the X-tree. Given an embedding, its dilation is the maximum distance in the X-tree between images of adjacent vertices of the tree. Our goal is to minimize the dilation, as the dilation corresponds to the number of clock cycles needed in the X-tree network to communicate between formerly adjacent processors in the tree. It is also important to minimize the size of the host network. The expansion of an embedding is the ratio of the size of the X-tree divided by the size of the tree.

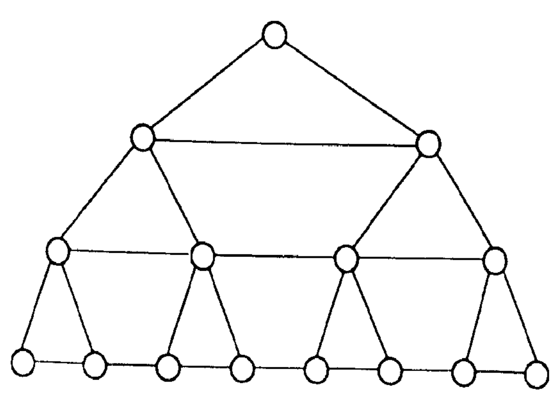

Figure 1: The X-tree of height 3

Often an embedding is not one-to-one. In this case also the load factor measures the quality of an embedding. The load factor is the maximum number of vertices of the tree mapped to any node of the X-tree. For networks of fixed size our goal is to minimize the load factor, as the load factor measures the computation work which has to be done by a single processor of the X-tree network.

Our main result is the following theorem, which shows that every binary tree can be embedded with dilation 3 and load factor 16 into its "optimal" X-tree.

Theorem 1: Let $T = (V, E)$ be an arbitrary binary tree with n nodes, $n = 16 \cdot (2^{r+1} - 1)$ for some r. Then there exists an embedding of T into the X-tree of height r with dilation 3 and load factor 16.

From Theorem 1 we can easily derive the following two theorems

Theorem 2: Let $T = (V, E)$ be an arbitrary binary tree with n nodes, $n = 16(2^{r+1} - 1)$ for some r. Then there exists an injective embedding of T into the X-tree of height $r + 4$ with dilation 11.

Theorem 3: Let $T = (V, E)$ be an arbitrary binary tree with n nodes, $n = 16 \cdot (2^r - 1)$ for some r. Then there exists an embedding of T into the hypercube Q_r with load factor 16 and dilation 4.

It had to be expected that the embedding into the hypercube found in theorem 3 by using the embedding from theorem 1 cannot match the specialized technique from [7] for embedding binary trees directly into the hypercube (with dilation 3 and constant expansion, and with dilation 5 without expansion, respectively). However, theorem 3 gives some new information. It shows that every binary tree can be embedded into its optimal hypercube with dilation 4 if we allow non-injective mappings with constant load factor.

A graph U with n nodes is said to be universal for a family G of n-node graphs if every graph in the family is a subgraph of U. This is a very strong simulation property since every computation on a network belonging to the family G can be simulated by U in real time. The problem of constructing minimal graphs for the family of all trees with the fewest number of edges has found considerable attention. In [4] and [5] it was shown that $0(n \cdot \log n)$ edges are necessary and sufficient. This result can be improved if we restrict ourselves to the family of binary trees. In [1], [2] and [6] it is shown how to construct a universal graph of bounded degree d, d being very large and left unspecified. We extend our embedding into the X-tree and construct a universal graph of "small" degree. This way we get a universal graph if the number of nodes is equal to $n = 2^i - 16$ for some i. We have no doubt that one could generalize this result to hold also for arbitrary n, but we have not done so in this paper.

Theorem 4: For every $n \in N$, such that $n = 2^i - 16$ for some i, there exists a graph G_n of degree bounded by 415 such that every binary tree with n nodes is a spanning tree of G_n.

Theorem 1 is proved in section 2 and the other results are proved in section 3.

2 The Proof of Theorem 1

In this section we will prove theorem 1. We start with a few definitions and two helpful lemmas about the separation of trees. The proofs of these lemmas are rather straightforward and a similar approach was used already in [7], but there are some details which are different from the formulation in [7] and which are important for the proof of theorem 1. This, we think, justifies to state the proofs also in this paper.

Let us recall the definition of an X-tree from [8].

Definition: The X-tree of height r, denoted by $X(r)$, is the graph whose nodes are all binary strings of length at most r and whose edges connect each string x of length $i(0 \leq i < r)$ with the strings xa, a in $\{0,1\}$, of length $i+1$ and, when binary$(x) < 2^i - 1$, also connects x with successor(x), where binary(x) is the integer x represents in binary notation and successor(x) denotes the unique binary string of length i such that binary(successor(x)) = binary$(x) + 1$. (For completeness let binary$(\epsilon) = 0$, where ϵ is the empty string).

Note that if we have given some tree $T = (V, E)$ and some set $S \subset V$ of nodes, then the graph $T_S = (S, \{\{u,v\} \in E; u,v \in S\})$, induced by S and T, is a forest. Let us denote this forest by $F(S,T)$.

Definition: S is called collinear with respect to T, or just collinear if T is understood, if any tree from $F(V-S,T)$ is connected by at most two edges to nodes from S.

Lemma 1: Let $T = (V,E)$ be an n node binary tree with two designated nodes r_1 and r_2. Let Δ be some natural number with $n > 4\Delta/3$. Then we can find two sets $S_1, S_2 \subset V$ with the following properties.

(1) $\{r_1, r_2\} \subset S_1 \cup S_2$

(2) $|S_1| \leq 4, |S_2| \leq 2$

(3) The deletion of the edges connecting nodes from S_1 with nodes from S_2 splits T into two forests T_1, T_2 with n_1, n_2 nodes, respectively, such that T_i contains all nodes from S_i for $i = 1,2$ and $|n_2 - \Delta| \leq \lfloor \frac{\Delta+1}{3} \rfloor$.

(4) S_i is collinear in T_i for $i = 1,2$.

Proof: Let T and Δ be as described above. For convenience we replace T with a directed tree T', containing the same vertices, but replace each edge $\{x,y\}$ of T by an edge connecting x and y and directed away from the designated node r_1. (In our proof, for ease of reading, T will denote the directed tree T'.) With directed edges we can refer without loss of generality, for any node z in T, to the subtree of T with root z, denoted by $T(z)$. Also, by $T(z,y)$ we denote the largest subtree of $T(z)$ that does not contain the vertex y.

First we consider the procedure find 1 which will find a node u with

$$(\lceil 4\Delta/3 \rceil - 1)/2 \leq |T(u)| \leq 4\Delta/3.$$

procedure find1 (u);
  while $|T(u)| > 4\Delta/3$ do
  let u be the child of u of maximal cardinality;

It is not difficult to verify that $||T(u)| - \Delta| \leq \lfloor(\Delta+1)/3\rfloor$ holds. Furthermore $r_1 \neq u$, since we have assumed that $n = |T(r_1)| > 4\Delta/3$ holds. Therefore we will define S_1, S_2 in such a way that $T_2 = T(u), T_1 = T(r_1, u)$ holds. We have to guarantee that S_1 andS_2 are collinear and we consider two cases. Let x be the father of u in T.

If $T(u)$ contains r_2 then we set $S_1 = \{r_1, x\}, S_2 = \{u, r_2\}$. If $T(r_1, u)$ contains r_2 then there exists some node y in $T(r_1, u)$ such that the path from r_1 to u and the path from r_1 to r_2 part at node y. Of course y may be equal to r_1 or equal to r_2, but in general y is a node different from r_1, r_2 and x.

In this case we set $S_1 = \{r_1, r_2, x, y\}, S_2 = \{u\}$. It is obvious that S_1 and S_2 are collinear. □

Lemma 2: Let T, n, r_1, and r_2 be as in Lemma 1 and let Δ be some natural number, $\Delta \leq n$. Then we can find two sets $S_1, S_2 \subset V$ which fulfill conditions (1) and (4) from lemma 1 and additionally,

(2') $|S_1|, |S_2| \leq 4$

(3') The deletion of the edges connecting nodes from S_1 with nodes from S_2 splits T into two forests T_1, T_2 with n_1, n_2 nodes, respectively, such that T_i contains all nodes from S_i for $i = 1, 2$ and

$$|n_2 - \Delta| \leq \lfloor \frac{\Delta + 4}{9} \rfloor$$

Proof: As in the proof of lemma 1 we assume that we have directed the edges in T away from the node r_1. Note that we can find a partition fulfilling condition (3') by applying procedure find1 twice. But we have to be a little bit more careful than in the proof of lemma 1 in order to guarantee the other conditions. First we assume that $|T| = n > 4\Delta/3$ holds. We start our algorithm by calling the following procedure find 2 with the argument v set to the designated node r_1:

procedure find 2 (v);
while $|T(v)| > 4\Delta/3$ and $v \neq r_2$ do
let v be the child of v on the path from r_1 to r_2;

This call computes a node v on the path from r_1 to r_2 such that either $|T(v)| \leq 4\Delta/3$ or $|T(v)| > 4\Delta/3$ and $v = r_2$. We consider three cases. In all these cases the condition $n > 4\Delta/3$ remains invariant during the computation.

1. $v = r_2$ and $|T(v)| > 4\Delta/3$
   In this case the designated nodes r_1 and r_2 are placed both into the set S_1. We find our partition by applying procedure find1 twice starting from node r_2.
2. $|T(v)| < \Delta$
   Let x be the father of v in the tree T. In this case the nodes r_1 and x are placed into set S_1 and the nodes r_2 and v into set S_2. We find our partition by applying procedure find1 twice in $T(x, v)$ starting from node x.
3. $\Delta \leq |T(v)| \leq 4 \cdot \Delta/3$.
   Let again x be the father of v in the tree T. The nodes r_1 and x are placed into the set S_1 and then the partition is used which is found in lemma 1 with the entries $T' = T(v), \Delta' = |T'| - \Delta$ and designated nodes $r_1' = v$ and $r_2' = r_2$.

We still have to consider the case $\Delta \leq n \leq 4\Delta/3$. In this case we solve the problem with $\Delta_1 = n - \Delta \leq \Delta/3$ and interchange the roles of S_1 and S_2 and of T_1 and T_2 afterwards. Note that $n \geq \Delta \geq 4 \cdot \Delta_1/3$ and therefore we can apply the algorithm described above. Furthermore $|n_2 - \Delta| = |n_1 - \Delta_1| \leq \lfloor(\Delta_1 + 4)/9\rfloor \leq \lfloor(\Delta + 4)/9\rfloor$. □

Now, we proceed to describe our embedding of an arbitrary binary tree into an X-tree with load factor 16, dilation 3 and optimal expansion. Note that any graph that is embeddable into an X-tree of height r with load factor 16 and optimal expansion has at most $16 \cdot (2^{r+1} - 1)$ nodes.

Theorem 1: Let $T = (V, E)$ be an arbitrary binary tree with n nodes, $n = 16 \cdot (2^{r+1} - 1)$ for some r. Then there exists an embedding into the X-tree of height r with dilation 3 and load factor 16.

Proof: The main idea of our construction is not very difficult, but we have to be very careful in describing it.

We define iterative partial embeddings $\delta_i : D_i \to X_i, D_i \subset V$, for $i = 1, \ldots, r$. For every i these embeddings will have the following properties:

(1) δ_i is an extension of δ_{i-1}; i.e. $D_{i-1} \subset D_i$ and $\delta_i(u) = \delta_{i-1}(u) \;\; \forall u \in D_{i-1}$.

(2) If $i < r$, then δ_i has load factor 16 and $|D_i| = 16 \cdot (2^{i+1} - 1)$; i.e. if $i < r$, then exactly 16 nodes of T are mapped onto every node of the X-tree X_i.

(3) δ_i has dilation 3; i.e. if $u, v \in D_i$ and $\{u, v\} \in E$, then there exists a path of length at most 3 connecting $\delta_i(u)$ and $\delta_i(v)$.

(4) If two nodes $u, v \in V$ are neighbors in T, then the levels of their images in the X-tree differ at most by an additive constant of two. I.e. let $u, v \in V$ with $\{u, v\} \in E$. Assume $u \in D_i$ and let $\delta_i(u)$ be a vertex in the X-tree on level j, $j \leq i - 2$. Then $v \in D_i$ holds and the level of the vertex $\delta_i(v)$ is some number j' with $|j - j'| \leq 2$.

First we will describe the construction informally. Let $R_i = V - D_i$ be the set of nodes of T not laid out so far. We attach every node from R_i to some leaf of X_i, i.e. we define a mapping $\rho_i : R_i \to \{0,1\}^i$. To every vertex α of the X-tree X_i we associate all the nodes of T which are mapped or attached to itself or to one of its successors in the X-tree, i.e. we set

$$A_i(\alpha) = \delta_i^{-1}(\alpha) \cup \rho_i^{-1}(\alpha) \quad \text{for } \alpha \in \{0,1\}^i$$
$$A_i(\alpha) = \delta_i^{-1}(\alpha) \cup A_i(\alpha 0) \cup A_i(\alpha 1)$$
$$\text{for } \alpha \in \{0,1\}^j, j < i.$$

Let us set $n_i = 16 \cdot (2^{i+1} - 1)$ for $i \in N$, i.e. n_i is the maximum number of nodes which can be embedded onto an X-tree of height i with load factor 16. In the final embedding δ_r we have of course $|A_r(\alpha)| = n_{r-|\alpha|}$ for all α. This is not true for values $i < r$, but our aim is to define the mappings δ_i and the attachments ρ_i in such a way that the differences $|n_{r-|\alpha|} - |A_i(\alpha)||$ get smaller and smaller for increasing values of i. We will try to get better approximations by going from the embedding δ_i to the embedding δ_{i+1} and we will use the horizontal edges on level $i+1$ of the X-tree X_{i+1} to obtain this improvement.

Furthermore we have to split $A_i(\alpha), |\alpha| = i$, into the sets $A_{i+1}(\alpha 0), A_{i+1}(\alpha 1)$, and we will use the edge $\{\alpha 0, \alpha 1\}$, to get good values for $|A_{i+1}(\alpha 0)|$ and $|A_{i+1}(\alpha 1)|$. Thus we use every horizontal edge on level $i + 1$ for one such adjustment.

To describe this construction more formally, let us consider $R_i = V - D_i$. Let F_i be the forest induced by R_i and T. Since T is connected, every tree from F_i is connected by at least an edge to some node from D_i. δ_i will have the following additional properties:

(5) D_i is collinear.

(6) If for some tree $\tilde{T} = (\tilde{V}, \tilde{E})$ from F_i there exist two different nodes $u, v \in D_i$ and $w_1, w_2 \in \tilde{V}$ with $\{u, w_1\}, \{v, w_2\} \in E$, then u and v are mapped by δ_i to the same vertex, i.e. $\delta_i(u) = \delta_i(v)$.

Thus, for every tree $\tilde{T} = (\tilde{V}, \tilde{E})$ from F_i the value $\delta_i(u)$ for any node $u \in D_i$ with $\{u, w\} \in E$ for some $w \in \tilde{V}$ is determined uniquely and will be denoted by $\alpha(\tilde{T})$. We will call $\alpha(\tilde{T})$ the <u>characteristic address</u> of $\tilde{T}$. Note that because of property (4) the characteristic address is a vertex on level $i - 1$ or on level i of the X-tree X_i.

As above, let $\tilde{T} = (\tilde{V}, \tilde{E})$ be a tree from F_i. Nodes $w \in \tilde{V}$ with $\{u, w\} \in E$ for some $u \in D_i$ are called <u>designated nodes</u> of $\tilde{T}$. Note that every tree from F_i contains at least one designated node and (because of property (5)) at most two designated nodes. Following the notation from [7] we call a tree with two designated nodes an <u>interval</u>. Furthermore we are building pairs of trees with the same characteristic address containing only one designated node. Such a pair of trees will also be called an <u>interval</u>. Note that this way to every vertex of X_i on levels $i - 1$ or i there are associated at most 16 intervals, since every node from D_i has at most 2 neighbors in R_i.

We will now use the characteristic addresses to define the attachment $\rho_i : R_i \to \{0,1\}^i$. All nodes of some tree $\tilde{T}$ are attached to the same vertex. If $\alpha(\tilde{T}) \in \{0,1\}^i$, then we set $\rho_i(u) = \alpha(\tilde{T})$ for all nodes u of $\tilde{T}$. If $\alpha(\tilde{T}) \in \{0,1\}^{i-1}$, then we set $\rho_i(u) = \alpha(\tilde{T})\beta(\tilde{T})$ for all nodes $u \in \tilde{T}$ and for some $\beta(\tilde{T}) \in \{0,1\}$.

Thus in order to define the attachment we need a mapping $\mu_i : \tilde{R}_i \to \{0,1\}$, where $\tilde{R}_i$ is the set of all nodes $u \in R_i$ for which there exists some node $v \in D_i$ with $\{u,v\} \in E$ and $|\delta_i(v)| = i-1$.

μ_i will fulfill the following properties:

(7) If two nodes $u, v \in \tilde{R}_i$ are neighbors of the same node $w \in D_i, |\delta_i(w)| = i-1$, then $\mu_i(u) \neq \mu_i(v)$.

(8) If two nodes $u, v \in \tilde{R}_i$ belong to the same tree in F_i then $\mu_i(u) = \mu_i(v)$.

The mappings δ_i and μ_i determine the embedding and the attachment and therefore also the sets $A_i(\alpha)$ for all $\alpha \in \{0,1\}^j, 0 \leq j \leq i$. In order to measure the quality of embedding and attachment we introduce the notations $nh(j,i), nl(j,i)$ and $\Delta(j,i)$ for $0 \leq j \leq i \leq r$.

Let $nh(j,i)$ and $nl(j,i)$, respectively, denote the maximal (and minimal, respectively) cardinalities of the set of nodes associated to any node on level j of the X-tree after i rounds. $\Delta(j,i)$ measures the maximal number of nodes which still have to be shifted between vertices on level j after i rounds. I.e.

$$\begin{aligned}
nh(j,i) &= \max\{|A_i(\alpha)|; |\alpha| = j\} \\
nl(j,i) &= \min\{|A_i(\alpha)|; |\alpha| = j\} \\
\Delta(0,i) &= 0 \\
\Delta(j,i) &= \tfrac{1}{2}\max_{|\alpha|=j-1} \big||A_i(\alpha 0)| - |A_i(\alpha 1)|\big| \\
&\quad \text{for } j > 0.
\end{aligned}$$

We are now ready to describe the construction of the embeddings $\delta_i, 0 \leq i \leq r$.

We start by defining δ_0. We choose some subtree $D_0 \subset V$ of 16 nodes and set $\delta_0(u) = \epsilon$ for all $u \in D_0$. All nodes from $R_o = V - D_o$ are attached to the vertex ϵ, i.e. $\rho_o(u) = \epsilon$ for all $u \in R_o$.

Now the embeddings $\delta_i, 1 \leq i \leq r$, are defined by the iterative algorithm X-TREE which is defined below.

algorithm X-TREE

```
for i := 1 to r do
begin
   for j := 0 to i - 2 do
      for all α ∈ {0,1}^j do
      ADJUST(α0, α1, i);
   for all α ∈ {0,1}^(i-1) do SPLIT(α, i)
end;
```

The procedures $ADJUST$ and $SPLIT$ are described in detail later. They determine which nodes from R_{i-1} are mapped to the leaves $\alpha, |\alpha| = i$, of X_i. Note that during round i we don't change the layout performed in the previous rounds and therefore δ_i is an extension of δ_{i-1}, i.e. condition (1) holds.

Both procedures $ADJUST$ and $SPLIT$ use the partition lemmas 1 or 2, respectively. The call $ADJUST(\alpha 0, \alpha 1, i), 0 \leq |\alpha| \leq i-2$, shifts one or two subtrees attached to the node $\alpha 0 1^{i-|\alpha|}$ to the node $\alpha 1 0^{i-|\alpha|}$ (or vice versa). Note that every vertex attached to node $\alpha 0 1^{i-|\alpha|}$ is also attached to $\alpha 0$ and every vertex attached to $\alpha 1 0^{i-|\alpha|}$ is also attached to $\alpha 1$ and therefore we can obtain this way values for $|A(\alpha 0)|$ and $|A(\alpha 1)|$ with a better balance. The call $SPLIT$ $(\alpha, i), |\alpha| = i-1$, partitions the set of trees attached to α into two sets which are attached now to $\alpha 0$ and $\alpha 1$. During these calls all the designated nodes defined by using the partitions from lemma 1 or lemma 2 are laid out. Also, during the call of procedure $SPLIT$ all nodes are laid our which are children of nodes laid out at level $i-2$ (if this has not been done before).

Note that this way 16 nodes are associated to every vertex of the X-tree. 4 nodes result from applying procedure $SPLIT$, 4 nodes from applying procedure $ADJUST$ and there may be 8 nodes which are children of nodes laid out in the grandparent vertex. Note that also 16 nodes are laid out in the grandparent vertex, which may have 32 children which are distributed among 4 vertices.

We can show that for $0 \leq j \leq i \leq r$

$$\begin{array}{ll} \Delta(i,i) \leq 2^{r+2-i} & \text{, if } i < r \\ \Delta(j,i) \leq 2^{r+j+1-2i} & \text{, if } j < i \text{ and} \\ & \quad 2i \leq r+j+1 \\ \Delta(j,i) = 0 & \text{, if } 2i \geq r+j+2 \end{array}$$

This implies that $\Delta(j,r) = 0$ for $j \leq r-2$ and the final embedding (i.e. $\Delta(j,r) = 0$ for all $0 \leq j \leq r$) can be obtained by some simple rearrangement in the last two levels.

The details, will be described in the following subsections:

(i) The procedure *ADJUST*

(ii) The procedure *SPLIT*

(iii) Estimations of $\Delta(j,i), nh(j,i), nl(j,i)$

(iv) Revision of the procedure *ADJUST*

(v) The final embedding

Because of lack of space the subsection (iv) and some details in (ii) and (iii) will not be described in this extended abstract.

While describing the procedures *ADJUST* and *SPLIT* we will also show that the embedding computed by our algorithm *X-TREE* fulfills conditions (2),...,(8). Instead of conditions (3) and (4) we will prove the slightly stronger condition (3').

(3') Let $u,v \in D_i$ with $|\delta_i(u)| \leq |\delta_i(v)|$. Then $\{u,v\} \in E$ implies that $\delta_i(v) \in N(\delta_i(u))$.

Here for each vertex α of the X-tree X_i let $N(\alpha)$ be the set of all vertices from X_i which can be reached from α by following a path in X_i consisting of at most three horizontal edges or of at most two downward edges followed by at most two horizontal edges. For the case $|\alpha| \leq i-2$, $\alpha \neq 00\ldots0, \alpha \neq 11\ldots1$, the set $N(\alpha)$ is shown in figure 2.

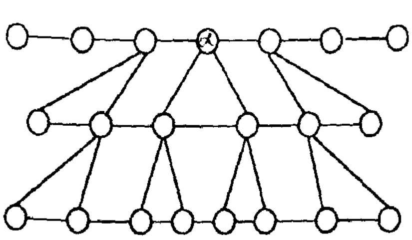

Figure 2: Some vertex α and its set $N(\alpha)$

It is clear, that condition (3) and condition (4) follow directly from condition (3').

(i) The procedure *ADJUST*

In the i-th loop of the algorithm the procedure *ADJUST* is called with the parameters $(\alpha 0, \alpha 1, i)$ for all $\alpha \in \{0,1\}^j$ and (in this successive order) for all $j = 0,1,\ldots,i-2$. Consider now some fixed j and some $\alpha \in \{0,1\}^j$. Let $A(\alpha 0)$ and $A(\alpha 1)$ be the sets of nodes associated to $\alpha 0$ and $\alpha 1$, respectively, when the algorithm calls $ADJUST(\alpha 0, \alpha 1, i)$.

Let $\Delta = \lfloor \frac{1}{2}(|A(\alpha 0)| - |A(\alpha 1)|) \rfloor$ be half the difference between $|A(\alpha 0)|$ and $|A(\alpha 1)|$ and assume *w.l.o.g.* that $|A(\alpha 0)| > |A(\alpha 1)|$ holds. Using lemma 2 we will now "shift" some nodes from $A(\alpha 0)$ to $A(\alpha 1)$ such that afterwards half the difference between $|A(\alpha 0)|$ and $|A(\alpha 1)|$ is at most $\lfloor (\Delta+4)/9 \rfloor$. In doing so we consider the sets of trees in F_{i-1} which are attached by ρ_{i-1} to the leaves $\alpha 01^{i-2-|\alpha|}$ and $\alpha 10^{i-2-|\alpha|}$.

First let us assume that in the set of trees attached to $\alpha 01^{i-2-|\alpha|}$ there exists some interval $\tilde{T}$ which has at least Δ nodes.

From the definition of the attachment we know that the characteristic address β of $\tilde{T}$ is either equal to $\alpha 01^{i-2-|\alpha|}$ or to the parent of $\alpha 01^{i-2-|\alpha|}$ in X_{i-1}. Now consider the two trees T_1 and T_2 obtained by splitting $\tilde{T}$ by using lemma 2.

We add the nodes from $S_1 \cup S_2$ to the domain of the embedding δ_i (which we are constructing during this loop) and we set

$$\begin{array}{ll} \delta_i(v) = \alpha 01^{i-1-|\alpha|} & \forall v \in S_1 \\ \delta_i(v) = \alpha 10^{i-1-|\alpha|} & \forall v \in S_2 \end{array}$$

Now let us assume, that all intervals from the set of intervals attached to $\alpha 01^{i-2-|\alpha|}$ have cardinality less than Δ, but that there exist two intervals I_1, I_2 with $|I_1| + |I_2| \geq 4\Delta/3$. Let $|I_1| \geq |I_2|$. Then $2\Delta/3 \leq |I_1| < \Delta$ holds. Let r_1, r_2 be the two designated nodes of I_1. First we shift the whole interval from $\alpha 01^{i-2-|\alpha|}$ to $\alpha 10^{i-2-|\alpha|}$. This is done by adding r_1 and r_2 to the domain of δ_i and by setting $\delta(r_1) = \delta(r_2) = \alpha 10^{i-1-|\alpha|}$. Afterwards half the difference between $|A(\alpha 0)|$ and $|A(\alpha 1)|$ is equal to $\Delta_1 = \Delta - |I_1| \leq \Delta/3$.

Now we apply lemma 1 with the interval I_2 and the value Δ_1. We can do the partition according to lemma 1 with a set S_2 of at most 2 elements. We add the elements from $S_1 \cup S_2$ to the domain of δ_i and set again

$$\delta_i(v) = \alpha 0 1^{i-1-|\alpha|} \quad \text{for } v \in S_1$$
$$\delta_i(v) = \alpha 1 0^{i-1-|\alpha|} \quad \text{for } v \in S_2$$

In this way we guarantee that $\lfloor \frac{1}{2}(|A_i(\alpha 0)| - |A_i(\alpha 1)|) \rfloor \leq \lfloor (\Delta + 4)/9 \rfloor$ holds and we have mapped at most 4 nodes from R_i to $\alpha 0 1^{i-1-|\alpha|}$ and 4 nodes to $\alpha 1 0^{i-1-|\alpha|}$

Note that $\Delta - \lfloor (\Delta + 4)/9 \rfloor \leq |T_2| \leq \Delta + \lfloor (\Delta + 4)/9 \rfloor$ and therefore after these shifts $\lfloor \frac{1}{2}(|A_i(\alpha 0)| - |A_i(\alpha 1)|) \rfloor \leq \lfloor (\Delta + 4)/9 \rfloor$ holds.

This call of $ADJUST$ laid out a few more nodes and we have to show that all the conditions remain valid. Conditions (2), (7) and (8) are not influenced by a call of $ADJUST$. Condition (5) remains valid since S_i is collinear in T_i for $i = 1, 2$, and condition (6) remains valid since all nodes from S_1 and from S_2, respectively, are mapped to the same vertex.

We still have to show that condition (3') is not affected by a call of $ADJUST$. Let us consider first the case where $\tilde{T}$ has at least Δ nodes.

Edges inside $S_1 \cup S_2$ connect nodes which are laid out at the same vertices or at adjacent vertices of the X-tree. So we have to consider only edges connecting nodes from $S_1 \cup S_2$ with nodes laid out earlier. Condition (5) holds and therefore at most 2 edges are connecting $\tilde{T}$ with D_{i-1}. These edges connect two nodes which are laid out at the characteristic address β of $\tilde{T}$ with the designated nodes of $\tilde{T}$. We just laid out these designated nodes (at the vertices $\alpha 0 1^{i-1-|\alpha|}$ or $\alpha 1 0^{i-1-|\alpha|}$) and since β is equal to $\alpha 0 1^{i-2-|\alpha|}$ or to the parent of $\alpha 0 1^{i-2-|\alpha|}$ in X_{i-1}, also these edges do not affect condition (3').

The second case, where there exist two intervalls I_1, I_2 with $|I_1| + |I_2| \geq 4\Delta/3$ can be studied now easily. First the designated nodes of I_1 are mapped to a leaf of the X-tree and this does not affect condition (3'). Afterwards lemma 1 is used and we can see just in the same way as above that condition (3') remains valid. □

(ii) The procedure SPLIT

In the i-th loop of the algorithm the procedure SPLIT is called with the parameters (α, i) for all $\alpha \in \{0,1\}^{i-1}$. Note that during the previous computations in this loop (i.e. by the calls of the procedure $ADJUST$) some extension of δ_{i-1} has been computed already and at most 4 nodes from R_{i-1} were mapped to each of the addresses $\alpha 0$ and $\alpha 1$. Then $A_i(\alpha)$ is the set of nodes associated to α during this previous computation.

Note that $A_i(\alpha)$ is given by the 16 nodes from D_{i-1} mapped already to α and by at most 28 intervals from F_{i-1}. These 28 intervals can be divided into three sets. There are at most 8 intervals whose characteristic address is equal to the father of α. Let S_1 be the set of these intervals. Their designated nodes have to be mapped now to $\alpha 0$ or $\alpha 1$, respectively. There are at most 16 intervals whose characteristic address is equal to α. Let S_2 be the set of these intervals. These intervals have to be attached now to $\alpha 0$ or $\alpha 1$, respectively. Finally there are 4 intervals which have been mapped provisionally to $\alpha 0$ or $\alpha 1$, respectively, during the computation of the procedure $ADJUST$. Let S_3 be the set of these intervals. These intervals may be shifted again from $\alpha 0$ to $\alpha 1$ or vice versa by the algorithm described below.

Let us consider again the set S_2. Note that the children of the designated nodes of an interval form one interval and two trees.

These two trees are combined logically to define a new interval. Now let I be any interval whose designated nodes have been mapped to the vertex α. Then one of its children will be attached to $\alpha 0$ and the other one to $\alpha 1$.

We perform the following algorithm in order to split $A_i(\alpha)$ into two sets M_0 and M_1.

$M_0 = M_1 = \emptyset$;
while $S_1 \cup S_2 \cup S_3 \neq \emptyset$ do
begin
  take two intervals I_1, I_2 from the same set S_1 or S_2 or S_3;
  w.l.o.g. $|I_1| \geq |I_2|$;
  add I_2 to the larger one of the two sets M_0

or M_1 and I_1 to the smaller one
end.

Here we can assume that each of the sets S_1, S_2, S_3 contains an even number of intervals. Otherwise we add an empty interval. If in S_2 there are still two intervals which are children of the same interval then choose one of them as I_1 and the other one as I_2.

In this way we guarantee that each one of the sets M_o and M_1 contains

- at most 4 intervals from S_1
- at most 8 intervals from S_2
- at most 2 intervals from S_3

and furthermore

- $\Delta = \frac{1}{2} \cdot (|M_o| - |M_1|)$ is bounded by the cardinality of the largest interval in $S_1 \cup S_2 \cup S_3$.

In order to get good estimations we have to associate the sets M_0 and M_1 to the vertices $\alpha 0$ and $\alpha 1$ in such a way, that the calls of *ADJUST* in the next round of our algorithm are influenced in a positive way. We will see in (iii) that we have to take special care about the call *ADJUST* $(\alpha 0, \alpha 1, i+1), |\alpha| = i-1$. Note that in the $(i+1)$st round each pair of vertices $\alpha 0, \alpha 1$ (except the special cases $\alpha = 0^{i-1}, \alpha = 1^{i-1}$) is influenced by exactly three calls of *ADJUST*. One of these calls is *ADJUST* $(\alpha 0, \alpha 1, i+1)$ and for each of $\alpha 0, \alpha 1$ there is one further call influencing it.

If we consider the case $\alpha = \hat{\alpha} 1 0^p, \hat{\alpha} \in \{0,1\}^*$, $p > 0$, then these two calls are *ADJUST* $(\alpha, successor(\alpha), i+1)$, *ADJUST* $(\hat{\alpha}0, \hat{\alpha}1, i+1)$. The first of these calls influences $\alpha 1$ and the second one influences $\alpha 0$.

Likewise if $\hat{\alpha} = \hat{\alpha} 0 1^p, \hat{\alpha} \in \{0,1\}^*, p > 0$, then these two calls are *ADJUST* $(predecessor(\alpha), \alpha, i+1)$ and *ADJUST*$(\hat{\alpha}0, \hat{\alpha}1, i+1)$.

Let Δ_0, Δ_1 be the two differences existing between the two pairs of trees that are adjusted. Then we associate M_0 and M_1 to $\alpha 0$ and $\alpha 1$ in such a way that the larger difference affects the larger set of nodes. I.e. in our first case $\alpha = \hat{\alpha} 1 0^p$ these two differences are

$\Delta_0 = \frac{1}{2}(|A_i(\hat{\alpha}1)| - |A_i(\hat{\alpha}0)|)$,
$\Delta_1 = \frac{1}{2}(|A_i(\alpha)| - |A_i(successor(\alpha))|)$

and M_0 is associated to $\alpha 0$ iff $\Delta_0 \geq \Delta_1$ and $|M_0| \geq |M_1|$ or $\Delta_0 < \Delta_1$ and $|M_0| < |M_1|$.

The nodes from S_1 and S_3 are laid out now. Note that at most 12 nodes are mapped to each of $\alpha 0$ or $\alpha 1$. The trees from S_2 are attached to $\alpha 0$ or $\alpha 1$, respectively, and in this way the mapping μ_i is defined.

The 4 free places in $\alpha 0$ and $\alpha 1$ are used now to reduce the difference between $A(\alpha 0)$ and $A(\alpha 1)$. Note that there exists a tree $\tilde{T}$ with at least Δ nodes. Therefore $\Delta \leq nh(i-1, i)$ and we can reduce the difference to $\Delta(i,i) \leq \lfloor (nh(i-1,i) + 4)/9 \rfloor$ by applying lemma 2.

If the number of nodes mapped to $\alpha 0$ (or $\alpha 1$, respectively) is smaller than 16 then the free places are filled by taking iteratively nodes which are attached to $\alpha 0$ (or $\alpha 1$, respectively) and which are not laid out so far, but which have at least one neighbour which has been laid out already.

Note that therefore condition (2) is fulfilled if the numbers $|A_i(\alpha 0)|, |A_i(\alpha 1)|$ of nodes attached to $\alpha 0$ and $\alpha 1$, respectively, are not smaller than 16. We will show in (iii) that for every $\beta \in \{0,1\}^i$

$$|A_i(\beta)| \geq nl(i,i) \geq n_{r-i} - a(i,i)$$

$$\geq 16(2^{r-i+1} - 1) - 2^{r+2-i} - 2^{r-i}$$
$$\geq 16(2^2 - 1) - 2^3 - 2 \geq 16$$

holds for $i < r$. Therefore condition (2) is fulfilled for $i < r$.

Because of lack of space we omit the proof that also the other conditions hold.

(iii) Estimations of $\Delta(j,i), nh(j,i), nl(j,i)$

We will show that for $0 \leq j \leq i \leq r$

$\Delta(i,i) \leq 2^{r+2-i}$, if $i < r$
$\Delta(j,i) \leq 2^{r+j+1-2i}$, if $j < i$ and $2i \leq r+j+1$
$\Delta(j,i) = 0$, if $2i \geq r+j+2$

and $nh(j,i) \leq n_{r-j} + a(j,i)$, $nl(j,i) \geq n_{r-j} - a(j,i)$,

where

$$\begin{array}{ll} a(i,i) \leq 2^{r+2-i} + 2^{r-i} & \text{, if } i < r \\ a(j,i) \leq 3 \cdot 2^{r+j-2i} & \text{, if } j < i \text{ and } 2i \leq r+j \\ a(j,i) \leq 1 & \text{, if } 2i = r+j+1 \\ a(j,i) = 0 & \text{, if } 2i \geq r+j+2 \end{array}$$

We will prove this assumption by induction on i. The assumption is true for $i = 0$, since

$$\Delta(0,0) = 0, \quad nh(0,0) = nl(0,0) = n_0 = n$$

Now let us assume $i \leq r$ and let the assumption be true for $0 \leq j \leq i-1$ and let us consider the i-th run of the algorithm.

First $A_{i-1}(0)$ and $A_{i-1}(1)$ are adjusted using lemma 2, therefore $\Delta(1,i) \leq \lfloor \frac{\Delta(1,i-1)+4}{9} \rfloor \leq \frac{\Delta(1,i-1)}{4}$.

Applying the procedure Adjust shifts some nodes between the forests $A_{i-1}(0)$ and $A_{i-1}(1)$. This influences also the trees $A_{i-1}(01), A_{i-1}(10), A_{i-1}(011), A_{i-1}(100)$, etc.

The cardinality of the forests associated to 01, 10, 011, 100, ... and therefore also the differences of brothers on the levels $j, 2 \leq j \leq i-1$, can be changed this way. Since half the difference between $A_{i-1}(0)$ and $A_{i-1}(1)$ before applying Adjust is at most $\Delta(1, i-1)$ and afterwards at most $\Delta(1,i)$, the cardinalities of the trees $T(01), T(10), \ldots$ are changed at most by $\Delta(1,i) + \Delta(1,i-1)$.

Note that in general every node in depth j can be influenced by at most one application of Adjust in some depth smaller than j. Since the differences $\Delta(j, i-1)$ are increasing in j, we know that after having applied the procedure Adjust to all vertices on levels smaller then j, the actual difference $\tilde{\Delta}(j, i-1)$ between siblings on level j is at most

$$\tilde{\Delta}(j,i-1) \leq \Delta(j,i-1) + \tfrac{1}{2}(\Delta(j-1,i) + \Delta(j-1,i-1) + \Delta(j-2,i) + \Delta(j-2,i-1)).$$

We now apply the procedure Adjust and get

$$\Delta(j,i) \leq \lfloor (\tilde{\Delta}(j,i-1) + 4)/9 \rfloor.$$

We have to compute $\tilde{\Delta}(j, i-1)$.

By the induction hypothesis, if $j < i-1$ and $2i \leq r+j-1$ then

$$\begin{aligned} \tilde{\Delta}(j,i-1) & \leq 2^{r+j+3-2i} + \tfrac{1}{2} \cdot (2^{r+j-2i} + 2^{r+j+2-2i} \\ & \quad + 2^{r+j-1-2i} + 2^{r+j+1-2i}) \\ & \leq 2^{r+j+3-2i} + \tfrac{1}{2} \cdot 2^{r+j-1-2i} \cdot (15) \\ & \leq 3 \cdot 2^{r+j+2-2i} \end{aligned}$$

Note that $\lfloor (6x+4)/9 \rfloor \leq x$ holds for all real numbers $x \geq 1$ and therefore $\Delta(j,i) \leq 2^{r+j+1-2i}$ follows for $1 \leq j < i-1$ and $2i \leq r+j-1$.

We have to consider also the remaining cases for $j < i-1$.

	$\tilde{\Delta}(j,i-1)$	$\Delta(j,i)$
$2i = r+j$	$\leq 8 + \frac{1}{2}(4+2+1) \leq 12$	$\leq \lfloor \frac{16}{9} \rfloor = 1$
$2i = r+j+1$	$\leq 4 + \frac{1}{2}(2+1) \leq 6$	$\leq \lfloor \frac{10}{9} \rfloor = 1$
$2i \geq r+j+2$	$\leq 2 + \frac{1}{2}(1) \leq 3$	$\leq \lfloor \frac{7}{9} \rfloor = 0$

The estimations of $\Delta(j,i)$ for $j = i-1$ and for $j = i$ and the estimations of the $a(j,i)$ cannot be described here because of lack of space.

(v) The final embedding

The mapping δ_r fulfills the following properties:

1. δ_r has dilation 3.
2. δ_r has mapped 16 nodes to any inner vertex of the X-tree X_r, and for every $\alpha \in \{0,1\}^r$ it has mapped 16 nodes to α iff $|A_r(\alpha)| \leq 16$ holds.
3. δ_r fulfills the estimations from (iii), especially $\Delta(j,r) = 0$, if $j \leq r-2$.

Therefore for every $\alpha \in \{0,1\}^{r-2}$ (defining a subtree of height 2), we can now distribute the nodes not laid our so far to free places among the leaves $\alpha00, \alpha01, \alpha10, \alpha11$. This embedding has still dilation 3 and therefore we have proved theorem 1. □

3 Proofs of Theorems 2,3,4

We can transform the embedding with load factor 16 from Theorem 1 in a straightforward way into an injective embedding.

Theorem 2: Let $T = (V, E)$ be an arbitrary binary tree with n nodes, $n = 16(2^{r+1} - 1)$ for some r. Then there exists an injective embedding of T into the X-tree of height $r + 4$ with dilation 11.

Proof: Let δ be the embedding into the X-tree X_r described in Theorem 1. δ has dilation 3 and load factor 16. We define an injective embedding χ into the X-tree X_{r+4} in such a way that for every $u \in V$

$$\chi(u) = \delta(u) \circ \mu$$

for some $\mu \in \{0,1\}^*, |\mu| = 4$.

For every address $\alpha, |\alpha| \leq r$, of the X-tree X_r there are exactly 16 nodes from T mapped onto α by δ. It is clear that we get an injective mapping χ if we use the 16 different addresses $\alpha\mu, |\mu| = 4$, as images of these 16 nodes. We do not have to specify this further in order to show that χ has dilation 11. Of course we have to use that δ has dilation 3.

Let $\alpha, \beta, \gamma, \omega$ be some nodes from X_r forming a path of length 3. Let μ and ν be some strings of length 4. We have to show that in X_{r+4} there exists a path of length at most 11 between $\alpha\mu$ and $\omega\nu$. Note that α and $\alpha\mu$ (and ω and $\omega\nu$, respectively) are connected by a path of length 4. Therefore $\alpha\mu - \ldots - \alpha - \beta - \gamma - \omega - \ldots - \omega\nu$ is a path in X_{r+4} of length 11. □

It is wellknown that a complete binary tree B_r can be embedded into its optimal hypercube Q_{r+1} with dilation 2 (see [8]). One way to establish this is the so called "inorder embedding", which is formally defined by

$$\delta_{io} : \cup_{i \leq r}\{0,1\}^i \rightarrow \{0,1\}^{r+1},$$

$\delta_{io}(\alpha) = \alpha 1 0^{r-|\alpha|}$ for $\alpha \in \{0,1\}^*, |\alpha| \leq r$. δ_{io} has dilation 2, since $\delta_{io}(\alpha 0) = \alpha 0 1 0^{r-1-|\alpha|}$, $\delta_{io}(\alpha 1) = \alpha 1 1 0^{r-1-|\alpha|}$ and therefore the image of the edge $\{\alpha, \alpha 0\}$ has dilation 2 and the image of the edge $\{\alpha, \alpha 1\}$ has dilation 1. Furthermore δ_{io} has the property that for any natural number λ, if α and β are nodes in B_r with distance λ, then $\delta_{io}(\alpha)$ and $\delta_{io}(\beta)$ have distance at most $\lambda + 1$ in Q_{r+1}.

The proof is straightforward. Let α and β have distance λ in B_r. Then there exists some $\gamma \in \{0,1\}^*$ and $\omega_1, \omega_2 \in \{0,1\}^*$ such that $\alpha = \gamma\omega_1, \beta = \gamma\omega_2$ and $\lambda = |\omega_1| + |\omega_2|$. Note that $\delta_{io}(\alpha) = \gamma\omega_1 1 0^{r-|\gamma|-|\omega_1|}$ and $\delta_{io}(\beta) = \gamma\omega_2 1 0^{r-|\gamma|-|\omega_2|}$ and therefore the distance between $\delta_{io}(\alpha)$ and $\delta_{io}(\beta)$ in Q_{r+1} is at most $\max\{|\omega_1|, |\omega_2|\} + 1$.

In a similar way X-trees can be embedded into hypercubes (see [8]). This construction has not been stated explicitly before and therefore we formulate it here as a lemma. Furthermore we need a special result for the construction we described in section 2.

Lemma 3: For any natural number r there exists an injective embedding δ of the X-tree X_r into the hypercube Q_{r+1} with the property that if α and β are nodes in X_r with distance λ, then $\delta(\alpha)$ and $\delta(\beta)$ have distance at most $\lambda + 1$ in Q_{r+1}.

Proof:
Our mapping $\delta : \cup_{i \leq r}\{0,1\}^i \rightarrow \{0,1\}^{r+1}$ is defined by $\delta(\alpha) = \chi(\alpha) 1 0^{r-|\alpha|}$, where for any $\alpha \in \{0,1\}^*, \alpha = a_1 \ldots a_i, a_u \in \{0,1\}$ for $1 \leq u \leq i$, $i \leq r$, we set

$\chi(\alpha) = b_1 \ldots b_i, b_u \in \{0,1\}$ for $1 \leq u \leq i$ with $b_1 = a_1$ and for $2 \leq \nu \leq i$: $b_\nu = a_\nu$ iff $a_{\nu-1} = 0$. Note that $|\chi(\alpha)| = |\alpha|$ for all α.

We show first that if α and β are siblings in X_r, then $\delta(\alpha)$ and $\delta(\beta)$ are neighbors in Q_{r+1}.
We assume w.l.o.g., that $\alpha \neq 1^{|\alpha|}$ and $\beta = successor(\alpha)$, i.e. $\alpha = \tilde{\alpha} 0 1^p$ with some $\tilde{\alpha} \in \{0,1\}^*, 0 \leq p < r$ and $\beta = \tilde{\alpha} 1 0^p$. We will see that $\chi(\alpha)$ and $\chi(\beta)$ differ from each other exactly in the $(|\tilde{\alpha}| + 1)$st bit.
If $p = 0$, then $\chi(\alpha) = \chi(\tilde{\alpha})b, \chi(\beta) = \chi(\tilde{\alpha})\bar{b}$, where $b \in \{0,1\}$ fulfills $b = 0$ iff the last bit of $\tilde{\alpha}$ is equal to 0.
If $p > 0$, then $\chi(\alpha) = \chi(\tilde{\alpha}0) 1 0^{p-1}, \chi(\beta) = \chi(\tilde{\alpha}1) 1 0^{p-1}$.

Now, let λ be some natural number and let α, β

two nodes in X_r of distance λ. Consider the th between α and β in X_r. Let λ_0 be the mber of horizontal edges on this path and let lenote the highest level reached on this path, . $p = \min\{|\gamma|; \gamma$ is a node on the path$\}$. Then ere exist $\gamma_1, \gamma_2, w_1, w_2 \in \{0,1\}^*$ with $|\gamma_1| =$ $| = p$, $\alpha = \gamma_1 w_1, \beta = \gamma_2 w_2, \lambda = \lambda_0 + |w_1| +$ $_2|$ and γ_1 can be reached from γ_2 by a path nsisting of λ_1 horizontal edges, $\lambda_1 \leq \lambda_0$.

ote that $\delta(\alpha) = \chi(\gamma_1)\tilde{w}_1 10^{r-p-|w_1|}, \delta(\beta) =$ $\gamma_2)\tilde{w}_2 10^{r-p-|w_2|}$ with some $\tilde{w}_1, \tilde{w}_2 \in \{0,1\}^*$, $_1| = |w_1|, |\tilde{w}_2| = |w_2|$.

ice $\chi(\gamma_1)$ and $\chi(\gamma_2)$ differ in at most λ_1 bits d since $\lambda_1 \leq \lambda_0$ holds, we see that $\delta(\alpha)$ and $\beta)$ differ in at most $\lambda_0 + \max\{|w_1|, |w_2|\} + 1$ s. □

om Lemma 3 and Theorem 1 the following the-m follows immediately.

ieorem 3: Let $T = (V, E)$ be an arbitrary iary tree with n nodes, $n = 16 \cdot (2^r - 1)$ for ne r. Then there exists an embedding of T o the hypercube Q_r with load factor 16 and ation 4.

is again a simple corollary from Theorem 3 it every binary tree with at most $2^r - 16$ nodes 1 be injectively embedded into the hypercube with dilation 8.

ir result about universal graphs follows also ily from Theorem 1. For $\alpha \in \{0,1\}^i, 0 \leq$ $\leq i$ let $N(\alpha)$ again be the set described in ure 2. Note that for very vertex α of an X-e the set $N(\alpha) - \{\alpha\}$ has at most 20 vertices d there exist at most 5 vertices β such that $\alpha \in$ $\beta)$ and $\beta \notin N(\alpha)$. Since our embedding from eorem 1 fulfills condition (3') it leads directly a universal graph of degree $25 \cdot 16 + 15 = 415$.

ieorem 4: For every $n \in N$, such that $n =$ $- 16$ for some i, there exists a graph G_n of gree bounded by 415 such that every binary e with n nodes is a spanning tree of G_n.

Acknowledgement:

The author wants to thank I.H. Sudborough for many helpful discussions and R. Klasing for a careful reading of this paper.

References:

1. S.N. Bhatt, F.R.K. Chung, F.T. Leighton, and A.L. Rosenberg, "Optimal Simulation of Tree Machines", Proc. 27th Annual IEEE Symp. Foundations of Computer Science, Oct. 1986, pp. 274-282.
2. S.N. Bhatt, F.R.K. Chung, F.T. Leighton, and A.L. Rosenberg, "Universal Graphs for Bounded-Degree Trees and Planar Graphs", SIAM J. Disc. Math. Vol. 2, 1989, 145-155.
3. S.N. Bhatt, F.R.K. Chung, J.-W. Hong, F.T. Leighton, and A.L. Rosenberg, " Optimal Simulations by Butterfly Networks", Proc. 20th Annual ACM Theory of Computing Symp., 1988, 192-204.
4. F.R.K. Chung and R.L. Graham, "On universal graphs", Proc. 2nd. Int. Conf. on Combin. Math., (A. Gerwirtz and L. Quintas, Eds.); Ann. N.Y. Acad. Sci. vo. 319, 1979, 136-140.
5. F.R.K. Chung and R.L. Graham, " On universal graphs for spanning trees", Journal London Math. Soc., 27, 1983, 203-211.
6. J. Friedman, N. Pippenger, "Expanding graphs contain all small trees", Combinatorica 7, 1987, 71-76.
7. B. Monien and I.H. Sudborough, "Simulating Binary Trees on Hypercubes", Proc. AWOC 88, LNCS 319, 1988, 170-180.
8. B. Monien and I.H. Sudborough, "Embedding one Interconnection Network in Another", Computing Supp. 7, 1990, 257-282.