Centrum Wiskunde & Informatica

REPORTRAPPORT

PNA

Probability, Networks and Algorithms

Probability, Networks and Algorithms

PNA

Embedding complete ternary tree in hypercubes using AVL trees

S.A. Choudum, I. Raman

REPORT PNA-E0812 DECEMBER 2008

Centrum Wiskunde & Informatica (CWI) is the national research institute for Mathematics and Computer Science. It is sponsored by the Netherlands Organisation for Scientific Research (NWO).
CWI is a founding member of ERCIM, the European Research Consortium for Informatics and Mathematics.

CWI's research has a theme-oriented structure and is grouped into four clusters. Listed below are the names of the clusters and in parentheses their acronyms.

Probability, Networks and Algorithms (PNA)

Software Engineering (SEN)

Modelling, Analysis and Simulation (MAS)

Information Systems (INS)

P.O. Box 94079, 1090 GB Amsterdam (NL)
Kruislaan 413, 1098 SJ Amsterdam (NL)
Telephone +31 20 592 9333
Telefax +31 20 592 4199

ISSN 1386-3711

Embedding complete ternary tree in hypercubes using AVL trees

ABSTRACT

A complete ternary tree is a tree in which every non-leaf vertex has exactly three children. We prove that a complete ternary tree of height h, TTh, is embeddable in a hypercube of dimension. This result coincides with the result of [2]. However, in this paper, the embedding utilizes the knowledge of AVL trees. We prove that a subclass of AVL trees is a subgraph of hypercube. The problem of embedding AVL trees in hypercube is an independent emerging problem.

2000 Mathematics Subject Classification: 68W10
1998 ACM Computing Classification System: G.2.2
Keywords and Phrases: Parallel algorithm, hypercube, complete ternary tree, AVL tree, embedding, dilation
Note: A part of second author‚Äôs work was carried out during her three months visit (May-July 2008) to Centrum Wiskunde en Informatica, PNA1 Group, 1098 SJ, Amsterdam, The Netherlands.

Embedding complete ternary tree in hypercubes using AVL trees

S.A. Choudum **Indhumathi Raman** *

Department of Mathematics
Indian Institute of Technology Madras
Chennai - 600036, India
email: sac@iitm.ac.in, indhumathi.raman@gmail.com

Abstract
A complete ternary tree is a tree in which every non-leaf vertex has exactly three children. We prove that a complete ternary tree of height h, TT_h, is embeddable in a hypercube of dimension $\lceil 1.6h \rceil + 1$. This result coincides with the result of [2]. However, in this paper, the embedding utilizes the knowledge of AVL trees. We prove that a subclass of AVL trees is a subgraph of hypercube. The problem of embedding AVL trees in hypercube is an independent emerging problem.

Keywords: Parallel algorithm, hypercube, complete ternary tree, AVL tree, embedding, dilation

1 Introduction

An embedding φ of a (guest) graph $G = (V_G, E_G)$ into a (host) graph $H = (V_H, E_H)$ is a map $\varphi : V_G \rightarrow V_H$ (not necessarily a bijection) such that every edge (u, v) of G is mapped onto a path P(φ(u), φ(v)) which connects φ(u) and φ(v) in H. The dilation of an edge (u, v) in G is defined to be the length of the path P(φ(u), φ(v)). The dilation of φ, dil(φ) is then defined to be the integer max{length (P(φ(u)), φ(v))) : (u, v) is an edge in G}. The expansion of φ is defined to be the ratio of V_H and V_G.

One of the motivations for these concepts is in the area of parallel algorithms and parallel computers. Here, the computational structure of a parallel algorithm A is represented by a graph G(A) and the interconnection network of a parallel computer N is represented by a graph H(N). An embedding of G(A) into H(N) describes the working of the parallel algorithm A when implemented on N. The dil(φ) is used to estimate the computational running time and the expansion of φ is used to estimate the number of unutilized processors in N. Leighton [9] gives an extensive survey of embedding various graphs into interconnection networks like hypercubes, meshes and tori. Note, that the dilation of any embedding is atleast 1, and if dil(φ) = 1, then G is (isomorphic with) a subgraph of H. In the following all our embeddings are injections, so the expansion is atleast 1. Clearly, an injective embedding with dilation 1 and expansion 1 is the one with least communication delay and the most cost effective. In this note, we first embed a subfamily of AVL trees into hypercubes, then use this result to give an alternate proof of a result of [2] which embeds complete ternary trees into hypercubes.

---

* A part of the author's work was carried out during her three months visit (May-July 2008) to Centrum Wiskunde en Informatica, PNA1 Group, 1098 SJ, Amsterdam, The Netherlands.

2 Preliminaries

Before we proceed to the embedding, we give a brief description of hypercubes and AVL trees.

2.1 Hypercubes

An n-dimension hypercube, Q_n, has 2^n vertices each labelled with a binary string of length n. Two vertices are adjacent if and only if their labels differ in exactly one position. We use the following properties of hypercubes for the embedding.

• Q_n is n-regular and its diameter is n.

• Q_n posseses exactly $2^n n!$ automorphisms which can be described as follows. For each subset S of {1, 2, … , n} and each permutation π of {1, 2, … , n}, the map $f(S, \pi) : V(Q_n) \rightarrow V(Q_n)$ defined by $f(S, \pi)(x_1 x_2 \ldots x_n) = (y_1 y_2 \ldots y_n)$ is an automorphism of Q_n where

$$y_i = \begin{cases} \overline{x}_{\pi(i)}, & \text{if } \pi(i) \text{ is in S} \\ x_{\pi(i)}, & \text{if othewise} \end{cases}$$

• Q_n is a vertex-symmetric, edge-symmetric and P_3-symmetric graph. (Here, P_3 is a path on 3 vertices)
A graph G is called a P_k-symmetric graph if for any two paths $P = (u_1, u_2, \ldots, u_k)$ and $Q = (v_1, v_2, \ldots, v_k)$, there exists an automorphism α of G such that $\alpha(u_i) = v_i$, for every i, $1 \leq i \leq k$. P_1-symmetric graphs and P_2-symmetric graphs are referred to as vertex-symmetric graphs and edge-symmetric graphs respectively.

2.2 AVL trees

An AVL tree is a rooted binary tree T in which for every vertex v in V(T), the heights of the subtrees, rooted at the left and right child of v, differ by at most one. Adelson-Velskii and Landis [1] defined these trees to provide most efficient data structures for computational routines like searching and sorting; see [3, 4, 8, 12, 13]. It is also known as height-balanced tree. Subsequently, there have been several variations of AVL trees all of which maintain some balance in the heights of the subtrees rooted at every internal node. All these variations are studied under a common title called the height balanced trees; see [5, 7, 11]. The importance of AVL trees is due to the fact that their height is logarithmic to their size. Therefore, operations like searching can be performed in logarithmic time on an AVL data structure, whereas the same operations can take linear time (in the worst case) on an arbitrary binary data structure. We are concerned with a subclass of AVL trees, T_h of height h which is defined as follows: $T_0 = K_1$, $T_1 = K_{1,2}$ and T_h ($h \geq 2$) is obtained by taking three copies of T_{h-2} with roots r_1, r_2, r_3 (say), and two new vertices R, S and adding the edges (R, S), (S, r_1), (S, r_2) and (R, r_3); see Figure 1 and R is designated as the root of T_h. The number of vertices (t_h) of T_h can be computed using the recurrence relation $t_h = 3t_{h-2}+2$, for $h \geq 2$ and the initial conditions $t_0 = 1$ and $t_1 = 3$.

On solving this relation, we get $t_h = \begin{cases} 2(3)^{\lfloor \frac{h}{2} \rfloor} 1, & \text{if h is even} \\ 4(3)^{\lfloor \frac{h}{2} \rfloor} 1, & \text{if h is odd} \end{cases}$

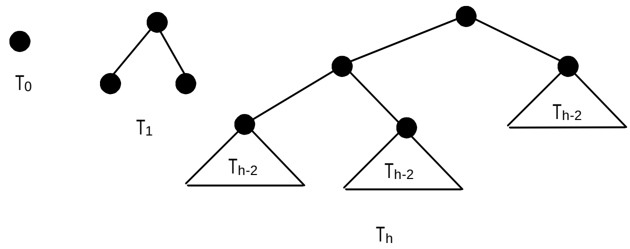

Figure 1: Structure of T_h

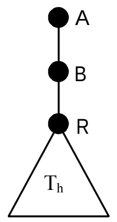

Figure 2: Structure of T_h^*

3 Embedding

3.1 Embedding AVL tree into hypercube

We are now ready to embed the AVL tree T_h (defined in section 2.2) into $Q_{d(h)}$ with expansion $2^{d(h)} / t_h$ which is close to 1. Thus the embedding is nearly optimal. The embedding of the AVL tree T_h is given by the following theorem.

Theorem 3.1: For every $h \geq 2$, T_h is a subtree of $Q_{d(h)}$ where

$$d(h) = \begin{cases} \left\lceil \frac{4h}{5} \right\rceil + 1, & \text{if } h \text{ is even} \\ \left\lceil \frac{4(h-1)}{5} \right\rceil + 2, & \text{if } h \text{ is odd} \end{cases}$$

Proof: To achieve such an embedding, we label T_h using d(h)-bit binary strings by recursion on h and depending on the value of h (mod 10). Since $Q_{d(h)} = Q_{d(h-2)} \times Q_t$ where $t = d(h) - d(h-2)$, we prefix every label of a copy of T_{h-2} by one of 2^t-bit string. We also make use of a small extension in the structure of the tree for the proof to work. Given a tree T_h with root R, we denote by T_h^*, a supertree of T_h formed by adding two new vertices A and B and two new edges (A,B) and (B,R). We call T_h^*, the auxillary tree of T_h and the path (A,B,R) the auxiliary path of T_h^*; see Figure 2.

3.2 Embedding complete ternary tree into hypercube

A complete ternary tree is a tree in which every non-leaf vertex has exactly three children. The structure of ternary trees has found application in parallel computing. In [10], the authors have developed and implemented software whose strategy was implemented using the CS tool software and a ternary tree network topology. A balanced ternary tree has been used to represent the widely used binary trees; see [12]. Hence, the problem of embedding ternary trees in hypercubes is of interest. In [6], Havel has conjectured the following:

Conjecture: The complete ternary tree can be embeddable into its optimal hypercube with dilation 2.

In this section, we first provide an embedding of a complete ternary tree of height h, TT_h in T_{2h} (discussed in section 2). We next provide an embedding of TT_h into the hypercube using the result of the previous section. The dimension of optimal hypercube for complete ternary tree is

$\lceil 1.585h \rceil + 1$. However in this paper, the expansion of our embedding is $\lceil 1.6h \rceil + 1$. Hence, there is a slight relaxation in the expansion factor as compared to the statement of the conjecture.

Theorem 3.2: A complete ternary tree TT_h of height h is embeddable into the AVL tree T_{2h} with dilation 2.

Proof: We prove by induction on h. Figure 3 gives a pictorial depiction of the embedding TT_h in T_{2h} with dilation 2.

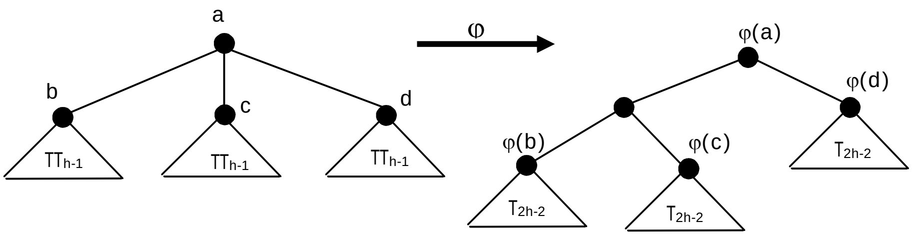

Figure 3: Embedding TT_h into T_{2h}

Combining Theorem 3.1 and Theorem 3.2, we have the following result.

Corollary 3.3: A complete ternary tree TT_h of height h is embeddable into a hypercube of dimension $\lceil 1.6h \rceil + 1$ with dilation 2.

The result has been proved by S.A. Choudum et. al., in [2]. The interesting aspect of the proof provided in this paper is the use of an AVL tree as an intermediate tree. The problem of embedding AVL trees in hypercubes is an independent research problem and is open.

4 References

[1] G.M. Adelson-Velskii and E.M. Landis, An algorithm for the organization of information, Soviet Math. Dokl., 3(1962), 1259-1262.

[2] S.A. Choudum and S. Lavanya, Embedding complete ternary trees into hypercubes, To appear in Discussiones Mathematicae Graph Theory.

[3] C.S. Ellis, Concurrent search and insertion in AVL trees, IEEE Trans. Comput., 29, 9(1980), 811-817.

[4] C.C. Foster, Information Storage and Retrieval Using AVL Trees, Proc. ACM 20th Nat. Conf.(1965), 192-205.

[5] C.C. Foster, A generalization of AVL trees, Comm. ACM, 16 (1973), 513 - 517.

[6] I. Havel, On certain trees in hypercubes, In Topics in combinatorics and graph theory, Physica-Verlag Heidelberg(1990), 353-358.

[7] P.L. Karlton, S.H. Fuller, R.E. Scroggs, and E.B. Kaehler, Performance of Height-Balanced Trees, Communications of the ACM, 19, 1(1976) 23 - 28.

[8] D.E. Knuth, Sorting and Searching, The Art of Computer Programming - 3, Addison-Wesley, 1973.

[9] F.T. Leighton, Introduction to Parallel Algorithms and Architectures: Arrays, Trees, Hypercubes, Morgan Kaufmann, San Mateo, CA, 1992.

[10] S. Liuni, N. Prunella, G. Pesole, T. D'Orazio, E. Stella and A. Distante, A new parallel algorithm for computation of statistically significant patterns in DNA sequences, Proceeding of the Twenty-Sixth Hawaii International Conference on System Sciences, 1(1993) 605 - 612.

[11] F. Luccio, L. Pagli, Power trees, Communications of the ACM, 21, 11 (1978), 941-947.

[12] K. Matsuzakshi and A. Morihata, Balanced Ternary-Tree Representation of Binary Trees and Balancing algorithms, Mathematical Engineering Technical Reports METR 2008-30. Available in www.ipl.t.u-tokyo.ac.jp/pub/METR2008-30.pdf.

[13] M. Medidi and N. Deo, Parallel Dictionaries using AVL-Trees, Journal of Parallel and Distributed Computing, 49(1998) 146 - 155.