5F－10

深さ優先探索技法を用いた２進木のハイパーキューブへの埋め込み実験

塩田 佳明, 渋沢 進

茨城大学工学部情報工学科

1　はじめに

これまでのハイパーキューブ結合に関するアルゴリズムは，規則正しいデータ構造を用いるものがほとんどであり，実際のプログラムに多く現れる不規則なデータ構造を扱ったものは少なかった．この不規則構造の中で，特に重要なものの１つに木構造がある．木構造は非常に多くの問題に適用され，重要な役割を演じる．しかし，任意の木構造のハイパーキューブへの埋め込みはNP完全であることが証明されている[1]．本研究では問題を限定して任意の２進木に着目し，ハイパーキューブの埋め込みに関して考察，実験を行った．アルゴリズムは深さ優先探索技法を採り入れたものであり，根から順にノードをたどって埋め込み，必要に応じてバックトラックを行う．

2　準備

グラフの埋め込みに関し，一般的に使われる用語を次のように定義する[2]．

dilation 埋め込み時に仮想的に伸ばした辺の長さのうち最大のもの．

expansion あるグラフ $G(V,E)$ から別のグラフ $G'(V',E')$ に埋め込むときの値 $n(V')/n(V)$ ．ここで，$n(V)$ はノード集合 V の要素数．

本論文で取り扱う任意の２進木は，根ノードは１または２の次数をもち，他のノードは1,2,3 のいずれかの次数をもつ．いわゆる完全２進木は根が２，葉が１，その他のノードが３の次数を持つ２進木である．

3　２進木埋め込みへの予想

埋め込みにおいて最も重要なことは木とハイパーキューブのノードの数のバランスを保つことである．すなわちdilationとexpansionを共にできるだけ小さくすることである．

図１に７ノード２進木の３次元ハイパーキューブへのdilation 1, expansion 1による埋め込み例を示す．木構造は根ノード付近でのデータの交換が一般的に多いので，この付近でdilation 2となることは望ましくない．逆にいえば，葉ノード付近でdilation 2となる時にはそれほど問題はないものと思われる．また，木のノード数が大きくなればなるほど，expansionの値は１に近いことが望ましい．なぜなら，埋め込んだ後に遊んでいるサブキューブの数も多くなるからである．しかし，共に１で埋め込むアルゴリズムはこれまでには見つかっておらず[3][4]，完全２進木のように埋め込むことができない木も存在する[5]．そこで我々はやや条件を緩め，任意の２進木は以下の条件のいずれかで埋め込めるのではないかという予想を立て，その実験による検証を試みた．

- dilation 1 かつ expansion 1
- dilation 1 かつ expansion 2
- dilation 2 かつ expansion 1

4　任意の２進木の生成

自然数 n に対して生成される任意の２進木の個数 T_n は，左右対称の木を許すとき以下のようになる．

補題　n を１以上の自然数とし，T_n をノード数 n の全ての２進木の個数とすると，T_n は n の偶奇により

$$T_n = \begin{cases} 2\sum_{i=0}^{n/2} T_i T_{n-i-1} & n\text{が偶数のとき} \\ 2\sum_{i=0}^{\lfloor n/2 \rfloor -1} T_i T_{n-i-1} + (T_{\lfloor n/2 \rfloor})^2 & n\text{が奇数のとき} \end{cases}$$

と表せる．ここで $T_0 = 1$ とする．　□

この式より T_n を計算すると，$n = 16$ で既に3000万個を越える木が生成される．そこで，左右対称である木を考えないことにすると，$n = 16$ では２万個程度に抑えることができるが，$n = 23$ で1000万個を越える．

5　埋め込み実験

深さ優先探索技法を用いたアルゴリズムにより実験を試みた．これは木を根から深さ優先で潜って行きながらノードを対応するハイパーキューブのノードに埋

An experiment embedding binary trees in a Hypercube using depth first search method

Yoshiaki SHIOTA, Susumu SHIBUSAWA
Faculty of Engineering, Ibaraki University

め込んで行き，埋め込めなくなったところでバックトラックを行ってまた別のノードに埋め込んで続行するという簡単なものである．アルゴリズムの概要を図2に示す．

このアルゴリズムを用いて実際に埋め込みの実験を行った．木のノード数が大きくなると，全ての木の組合せに対する実験は困難になるので，左右対称であるものを除いたノード数7から16までの全ての木の組合せに対し，ノード数7,8の木はノード数8の，それ以外の木はノード数16の，それぞれ3,4次元ハイパーキューブへのdilation 1かつexpansion 1による埋め込みを行い，予想の検証を試みた．

実験の結果を表1に示す．ここでnは木のノード数，T'_nは左右対称であるものを除いたnノード2進木の個数，D1E1はdilation 1かつexpansion 1で埋め込めた2進木の個数であり，max-bt, btはそれぞれ埋め込みが成功した木のバックトラックの回数の中で最大のもの，平均バックトラック回数を表す．また，バックトラックの回数は60回で打ち切ってある．

例を挙げると，7ノード木は24通り中22の木がdilation 1かつexpansion 1の埋め込みに成功している．埋め込めなかった木の1つは完全2進木で，もう1つは完全2進木に同形な木である．

6 考察

同じdilation 1でも，木のノードとハイパーキューブのノードの数の差が大きい場合と小さい場合では，埋め込める確率が大きく異っている．

ノード数が大きい時にはバックトラックの平均も大きくなっているが，これは一部の激しいバックトラックを生み出す木のためであり，実際には埋め込めた木のほとんどがバックトラックの回数は10以下であった．

また，今回はバックトラックの数を制限したが，これを増やせば埋め込める木の数を増加させることは可能である．

7 おわりに

今後は，激しいバックトラックを生むような2進木をどう取り扱うかを吟味していく予定である．

参考文献

[1] A.Wagner and D.G.Corneil: *Embedding Trees in a Hypercube Is NP-Complete*, SIAM J.Comput., vol.19, No.3, pp.570-590 (1990).

[2] F.T.Leighton: *Introduction to Parallel Algorithms and Architectures. Arrays. Trees. Hypercubes*, Morgan Kaufmann (1992).

[3] A.Wagner: *Embedding Arbitary Binary Trees in a Hypercube*, J.Parallel Distributed Comput., 7, pp.503-520 (1989).

[4] S.N.Bhatt, F.R.Chung, F.T.Leighton and A.L.Rosenberg: *Efficient Embeddings of Trees in Hypercubes*, SIAM J.Comput., Vol.21, No.1, pp.151-162 (1992).

[5] 永田 元康: 並列処理・分散処理, コロナ社 (1994).

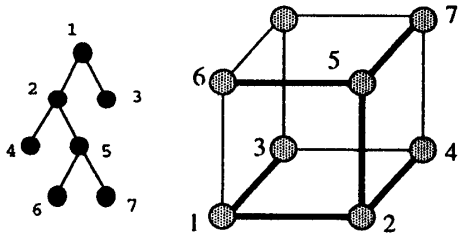

図1: 7ノード2進木と埋め込みの例

```
procedure tree-embedding
  pointer ← root;
  root に訪問済みの印をつける;
  cube ← 0**0;
  while(1)
   if(ポインタが左の子を持ち，それが未訪問) then
     隣接 cube の中で，まだ使われていないものの中で
     接続リンクの次元が最小の cube ← 左の子;
     if (上記のものが存在しない) then
          バックトラックを行う;
     else 左の子ノードに訪問済みの印をつける;
          pointer ← 左の子;
          cube ← 選ばれた cube のアドレス;
   else if(ポインタが右の子を持ち，それが未訪問
) then
       右の子と同様の処理;
   else 親ノードに戻る;  /* 訪問済だった */
```

図2: 埋め込みアルゴリズムの概要

表1: アルゴリズムの実行結果

n	T'_n	D1E1	max-bt	bt
7	24	22	2	0.36
8	47	27	4	0.74
9	103	103	4	0.07
10	214	214	4	0.16
11	481	481	6	0.29
12	1030	1015	26	0.51
13	2337	2294	48	1.12
14	5131	4732	54	2.14
15	11813	8714	57	3.82
16	26329	9838	59	6.82