ショートノート

建築図面データによる GBD 木の改良方式の性能評価

下平 丞作士†

GBD 木とその改良方式を建築図面データに適用し，数値実験によりその性能評価を行ったものである．改良方式では，大きな図形については，外接長方形を一定の大きさごとに分割して生成したサブ外接長方形を用いて木を形成している．実験の結果，次のことが分かった．GBD 木とその改良方式は，かなり規模の大きなデータを高速で処理することができる．改良方式では，長い線分については，スパン長程度の寸法で外接長方形分を割すれば，所要メモリ量が増えるが，削除と検索の性能が向上する．挿入時の処理時間は増える場合も減る場合もある．

Performance Tests on the GDB Tree Employing an Improved Method by Using Architectural Drawing Data

Hisashi Shimodaira†

In this paper, performance tests on a GBD tree and its improved version using architectural drawing data are performed. In the improved version, the minimal bounding rectangle (MBR) of a graphic element with large extent is divided into sub-minimal-bounding-rectangles (sub-MBR) and the tree is constructed by using these sub-MBR's. The experiments showed the followings. A GBD tree and its improved version have high performance. Comparing the improved version with the original one, when dividing MBR's of long line segments into sub-MBR's with about dimension of the span length of the building, the delete and search performance improves considerably, while the memory required for the tree structure data increases a little. The insert performance depends on cases.

1. まえがき

GBD 木[1]は，近くに存在する図形同士を階層的にグループ化して木構造で管理する図形データ管理手法の一種である．この方式では，図形の挿入・削除時には，図形の図心の位置をインデックスとして用いて近くに存在する図形同士をグループ化し，能率よく木構造を形成する．検索時には，図形を包含する外接長方形をインデックスとして用いているが，外接長方形の重なりについては特別の対策を講じていないので，比較的小さな図形を扱うのに適している．大きな図形が混在する場合には，外接長方形相互の重なりが増加するため，検索性能が低下する[2]．

筆者ら[2]は，大きな図形については，その外接長方形を一定の大きさ（Dmax）ごとに分割して生成したサブ外接長方形を用いて GBD 木を形成することにより，外接長方形相互の重なりを少なくし，検索性能の向上を図った改良方式を提案した．乱数により生成した線分データを用いた数値実験によると，適切な Dmax を用いればかなり検索性能が向上するが，一方で，挿入時の処理時間と木構造データについての所要メモリ量が増えるということが分かった．適切な Dmax の値は，扱う図形の性質（対象平面と図形の大きさの関係，大きな図形の量等）に依存するため，適用分野のデータについて数値実験を行って目安をつけることが必要である．

GBD 木およびその改良方式の性能は，乱数によって発生させた線分データを用いて検証されているが[1),2)]，建築図面データへの適用の事例はない．建築図面は，基準線に基づいて構成部品の位置が表現されており，基準線とこれらの構成部品を表す図形から構成されている．そのデータとしての特徴は次のとおりである．数は全体の数%であるが，対象平面の辺長と同程度の長さの基準線を含んでいる．バルコニー・間仕

† 日本メックス(株) FM 支援サービス事業部
FM Service Department, Nihon MECCS Co., Ltd.

切り壁等は，スパン長（隣合う基準線間の距離）の数倍程度の寸法である．壁等は，スパン長程度の寸法である．柱・家具等は，スパン長の数分の1程度の寸法である．したがって，建築図面データにGBD木を適用した場合，基準線やバルコニー等の長い線分の外接長方形が検索性能に悪影響を及ぼすことが予想される．このような性質のデータには改良方式が適していると考えられるが，実際に適用してみて，性能の改善効果を検証する必要がある．

本論文の目的は，建築図面データにGBD木およびその改良方式を適用し，数値実験によりその性能評価を行い，併せて，改良方式については適切な $D\max$ の値の目安を提示することにある．

2. GBD木とその改良方式の概要

GBD木の原方式[1]では，図形の挿入時には，図形の位置をその図心で表し，挿入する図形の位置と数に応じて対象平面の2等分割を繰り返して領域を生成する．各図形は，その図心が存在する領域に所属するようにグループ化し，木構造を形成する．原方式によるGBD木の例を，**図1**に示す．図形の外接長方形は細い実線で，領域は太い実線で示してある．この例では，図形 A, C, E は領域 R1 に，D, B は R2 に所属している．検索時には，各図形の外接長方形と各領域に存在するすべての図形を包含する外接長方形（図1では破線で示してある）をインデックスとして用いる．図形の外延が領域をはみ出しているときは，その外接長方形が互いに重なることとなる．図1では，図形Eの外延が領域R2にはみだしているため，領域R1とR2の外接長方形はかなり重なっている．重なっている部分を探索する場合，重なっているすべての領域を探索しなければならない．大きな図形が数多く存在するほど，領域の外接長方形相互の重なりが多くなり，検索性能が低下することとなる．

改良方式[2]では，図形の外接長方形の限界値（$D\max$）を与え，外接長方形の辺長がこの値以上のときは，辺長が $D\max$ 以下になるようなサブ外接長方形が生成し，もとの図形の外接長方形の代わりに用いる．サブ外接長方形は，もとの図形の外接長方形の各辺を $D\max$ で区切ったときの数と端数を考慮して，各辺を等分割して生成する．このとき，サブ外接長方形のうち，図形の一部を包含するもののみを登録する．大きな図形については，図形の挿入時に上記のようにしてサブ外接長方形を生成し，サブ外接長方形の図心の位置に基づいて，その個数分だけ挿入処理を行う．サブ外接長方形を格納した葉ノードのスロットには，もとの図形を指すポインタをもたせる．検索時には，サブ外接長方形を用いてもとの図形を検索する．改良方式によるGBD木の例を，**図2**に示す．この例では．図形Eの外接長方形はE1とE2に分割されて，領域R1とR2に分かれてグループ化されている．この方式によれば，大きな図形の外接長方形は実質的に小さくなる場合が多く，木の形成時にはサブ外接長方形ごとに近くの図形とグループ化され，各領域の外接長方形が小さくなるために，結果として領域の外接長方形相互の重なりが少なくなり，検索性能が向

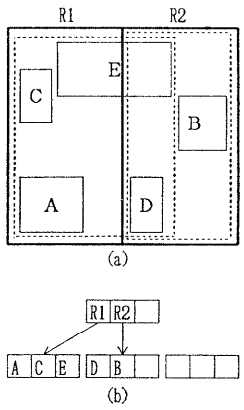

図 1 GBD 木の例（原方式）
（a）図形の外接長方形と領域，（b）木構造
Fig. 1 Example of GBD-tree (Original method).
(a) Rectangles organized into GBD-tree,
(b) GBD-tree for rectangles in (a).

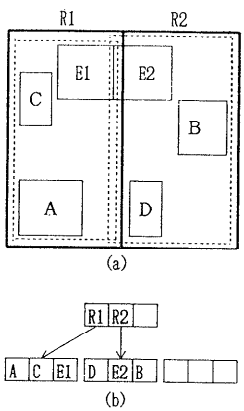

図 2 GBD 木の例（改良方式）
（a）図形の外接長方形と領域，（b）木構造
Fig. 2 Example of GBD-tree (Improved method).
(a) Rectangles organized into GBD-tree,
(b) GBD-tree for rectangles in (a).

上することとなる.

3. 性能テスト

3.1 テスト環境およびテスト方法

GBD 木の原方式と改良方式をインプリメントし，数値実験により各種性能の評価を行った．使用した計算機は，NEC PC-9801RA5 (80387 数値演算コプロセッサを使用) であり，MS-DOS Ver. 3.30C の環境下で，C言語により，図形の実データと木構造のデータをすべて主記憶に格納するようにインプリメントした．

テストは文献2)と同様な方法で行った．まず，あらかじめ主記憶に読み込んでおいた CAD データについて GBD 木を構築し，その際の残りの10% の図形をランダムに挿入する際の CPU タイムを計測した．ついで，長方形の検索範囲により，全体の5%の図形を検索する際の CPU タイムを計測した．最後に，全体の10% の図形をランダムに削除する際の CPU タイムを計測した．

3.2 テストデータ

建築図面の種類には，平面図，立面図等があるが，そのデータとしての性質は同じであるため，ここでは，平面図を対象とする．**図3**にテストデータの1例を示す．この例は，外周にバルコニーを有する一般的な平面の建物であり，X 方向14スパン（スパン長7 m)，Y 方向8スパン（スパン長7 m）である．データサイズ(図形数：M)は，基準線24本を含めて1000である．なお，ここでは柱等を表す長方形を一つの図形として扱っている．図面データが存在する対象平面は，辺長が 128 m の正方形領域としている．

原方式と改良方式により，**表1**に示す3シリーズのデータについてテストを行った．シリーズ1では，図3のデータを使用した．改良方式については，Dmax の影響を調べるために，スパン長の約1/2, 1, 2, 4, 7倍の5種類の Dmax を用いてテストを行った．シリーズ2では，データの角度の影響を調べるために，図3のデータを水平軸から 15, 30, 45 度傾斜させたデータを作成してテストした．シリーズ3では，データサイズの影響を調べるために，図3のデータに机等を表す長方形のデータを追加し，M=2000, 3000, 4000のデータを作成してテストした．M=1000のデータは基本図面に相当し，Mが2000以上のデータは建物の使用時に相当する．対象とする建物は平面的にかなり規模が大きいため，これらのデータもかなり規模の大きなものに属する．なお，シリーズ2と3では，シリーズ1の結果に基づいて，Dmax=8m を使用している．

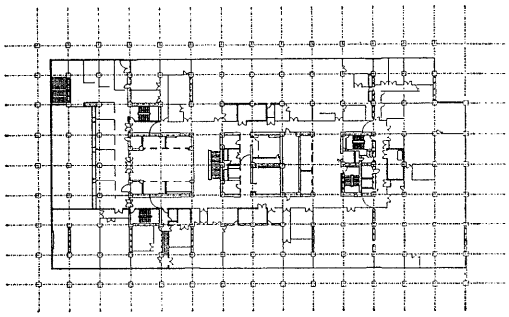

図 3 テストデータの例 (M=1000)
Fig. 3 An example of test data (M=1000).

3.3 テスト結果

各テストシリーズについては，挿入・削除・検索時の1図形（改良方式による検索では，サブ外接長方形ではなく本来の1図形）当りの CPU タイム，検索時の1図形当りのタッチノード数（検索時にタッチしたノード数を該当図形数で除したもの)，および木構造データについての所要メモリ量を，**表2**，**表3**，および**表4**に示す．これらの結果から，総じて，GBD 木およびその改良方式は，かなり規模の大きなデータを高速で処理することができることが分かる．原方式と比較した改良方式の性能の主な特徴は，次のとおりである．

テストシリーズ1（表2）：Dmax が小さくなるにつれて，挿入・削除・検索の性能（CPU タイム，タッチノード数）がともに向上しているが，挿入と検索

表 1 テストデータの内容および適用方式
Table 1 Contents of test data and applied method.

No.	Contents of data	Applied method	Results
1	M=1000 Change Dmax: 4, 8, 16, 32, 48 m	O.M. and I.M.	Table 2
2	M=1000 Change Angle: 0, 15, 30, 45 deg.	O.M. and I.M. (Dmax=8 m)	Table 3
3	Change M: 1000, 2000, 3000, 4000	O.M. and I.M. (Dmax=8 m)	Table 4

O.M.: Original method. I.M.: Improved method.

表 2 Dmax と性能の関係 (M=1000)
Table 2 Relation between Dmax and performance (M=1000).

Measured item	Original method	Improved method Dmax: 4	8	16	32	48
CPU time per element for insert. (msec.)	3.40	4.40	2.70	2.40	3.52	3.34
CPU time per element for delet. (msec.)	5.96	5.24	5.44	5.48	5.98	5.58
CPU time per element for search (μsec.)	140	113	83	98	99	122
Node touched per element for search	0.306	0.245	0.163	0.204	0.204	0.265
Memory required for tree data (kByte)	45.9	68.4	52.7	48.3	47.4	45.4

表 3 データの角度と性能の関係 (M=1000)
Table 3 Relation between data angle and performance (M=1000).

Data angle	CPU time per element for insert. (msec.)		CPU time per element for delet. (msec.)		CPU time per element for search (μsec.)		Node touched per element for search		Memory required for tree data (kByte)	
(degree)	O.M.	I.M.	O.M.	I.M.	O.M.	I.M.	O.M.	I.M.	O.M.	I.M.
0	3.40	2.70	5.96	5.44	140	83	0.306	0.163	45.9	52.7
15	3.36	2.52	8.30	4.68	166	132	0.340	0.255	45.4	53.2
30	3.22	3.58	8.60	4.68	213	142	0.435	0.283	44.9	57.6
45	3.72	3.88	5.28	5.26	233	155	0.500	0.326	43.9	58.6

表 4 データサイズと性能の関係
Table 4 Relation between data size and performance.

Data size	CPU time per element for insert. (msec.)		CPU time per element for delet. (msec.)		CPU time per element for search (μsec.)		Node touched per element for search		Memory required for tree data (kByte)	
	O.M.	I.M.	O.M.	I.M.	O.M.	I.M.	O.M.	I.M.	O.M.	I.M.
1000	3.40	2.70	5.96	5.44	140	83	0.306	0.163	45.9	52.7
2000	2.66	3.18	7.02	5.75	113	89	0.221	0.173	78.6	89.8
3000	3.22	4.55	6.68	5.13	86	64	0.154	0.115	115.7	134.3
4000	3.58	3.67	5.35	5.30	85	72	0.165	0.149	153.8	168.5

については，Dmax が 16 m と 8 m で改善効果は頭打ちになっている．これは，サブ外接長方形の個数の増加とこれに伴う木の高さの増大によるものである．原方式に比べて，挿入時の CPU タイムは Dmax=16 m で29%減，検索時の CPU タイムとタッチノード数は Dmax=8m でそれぞれ41%減と47%減，木構造データについての所要メモリ量は15%増である．

テストシリーズ2（表3）：角度が30度と45度の場合の挿入時の CPU タイムを除いて，改良方式の挿入・削除・検索の性能は，すべての角度について原方式よりも優れている．削除時の CPU タイムは30度で46%減となっている．木構造データについての所要メモリ量は，45度で33%増である．

テストシリーズ3（表4）：データサイズが大きくなると，挿入時の CPU タイムは改良方式の方が大きい（データサイズ3000の場合，41%増）．削除と検索の性能は，改善の程度は変動するが，常に改良方式の方が優れている．データサイズが大きくなると，検索性能の改善効果が小さくなっている．これは，サイズの大きなデータは M=1000 のデータに比較的小さな図形を追加して作成しているため，長い線分の比率が小さくなっているためである．

4. むすび

本論文では，GBD 木とその改良方式を建築図面のデータに適用し，その性能評価を行った．その結果，次のことが分かった．GBD 木とその改良方式は，かなり規模の大きなデータを高速で処理することができる．改良方式では，Dmax としてスパン長程度の値を用いれば，削除と検索の性能が向上する．ただし，長い線分の比率が小さい場合には，性能改善の効果は小さくなる．挿入時の性能は，向上する場合も，低下する場合もある．乱数を用いて発生させた線分データによるテスト結果[2]と比べると，検索性能のみでなく，削除性能と場合により挿入性能も向上している点が注目に値する．Dmax として，建物のスパン長という

物理的な寸法を用いて，その性能をコントロールできることも改良方式の特徴である．GBD木はR木[3]に比べて，検索性能はほぼ同等で挿入時の処理時間は2分の1から3分の1である[1]．上記の結果から，建築図面データに適用した場合には，改良方式の GBD 木はR木に比べて優れた性能を有していると言えよう．

謝辞 本研究は，筆者が NTT 在職中に行ったものである．研究の機会を与えていただいた関係者の方々，ご助言をいただいた東京大学生産技術研究所坂内正夫教授，埼玉大学工学部情報工学科大沢裕助教授に感謝の意を表します．

参 考 文 献

1) 大沢 裕，坂内正夫：2種類の補助情報により検索と管理性能の向上を図った多元データ構造の提案，電子情報通信学会論文誌（D），Vol. J74-D-I, No. 8, pp. 467-475 (1991)．
2) 下平丞作士，大沢 裕，坂内正夫：GBD 木の検索性能の改良法—大きな図形を扱うための手法の提案，情報処理学会論文誌，Vol. 33, No. 10, pp. 1254-1262 (1992)．
3) Guttman, A.: R-trees: A Dynamic Index Structure for Spatial Searching, *Proc. of ACM SIGMOD*, Vol. 14, No. 2, pp. 47-57 (1984)．

（平成4年 7 月31日受付）
（平成4年11月12日採録）

下平丞作士（正会員）

昭和16年生．昭和44年東京都立大学工学部建築工学科卒業．昭和46年同大学院修士課程修了．同年日本電信電話公社（現株式会社）入社．武蔵野電気通信研究所および建築部建築技術開発室において，数値解析・CAD・データベース・AI などの基礎理論とシステム開発に従事．平成4年から日本メックス(株)において，建物・設備管理用システムの研究・開発に従事．CAD，建物・設備管理用システムなどへの AI 技術の応用に興味を持っている．工学博士．日本建築学会，電子情報通信学会各会員．