ＬＳＩレイアウトシステム用データ構造　－ Bucket Tree －

3V-5

岩崎博昭　　三橋　隆
東芝　　ＵＬＳＩ研究所

１．緒　言

ＬＳＩの大規模化・高集積化にともなってレイアウトシステムへの要求も高まりつつあり、特に大量（データ数10^5 以上）のレイアウトデータを高速に処理する必要が生じてくるため、効率的なアクセスが可能なデータベースを開発することが必要不可欠となる。

本レイアウトシステムでは効率の良い会話的処理を可能とするグラフィック・エディタ（ＧＥ）の開発を目的として、指定された位置からレイアウトデータを高速にアクセスできるデータ構造の検討を行ってきた。本報告では、新しいデータ構造と現在までに提唱されているいくつかのデータ構造との性能を比較し、評価を行った結果について報告する。

２．Ｂｕｃｋｅｔ　Ｔｒｅｅ

今回の評価に用いたデータ構造は図１に示したような構成となっている。まず、対象とする領域（Root）を縦方向の分割線（Ｖ）によって分割し、各図形データが左（または右）の領域に完全に含まれる場合は左（または右）のこノードに所属させ（1, 5, 6 ）、分割線に交差するもの（2, 3, 4 ）についてはBisector List として格納する。分割線はルートから始まるツリーの深さに対応して２の剰余系（0, 1 ）で与えられるキーが０の時は縦方向、１の時は横方向とする。

分割の終了を判断する条件については、領域の面積が一定の値以下になったら終了する手法と領域内の図形数で判断する手法の比較を行った。

Bisector List は図２のように、分割線が縦方向の場合図形データの下辺と上辺を基準とし、横方向の場合は左辺と右辺を基準とした２次元のK-d Treeで構成される。この手法により、K-d TreeとQuad Tree の利点をとりいれたデータ構造を構成することが可能となる。

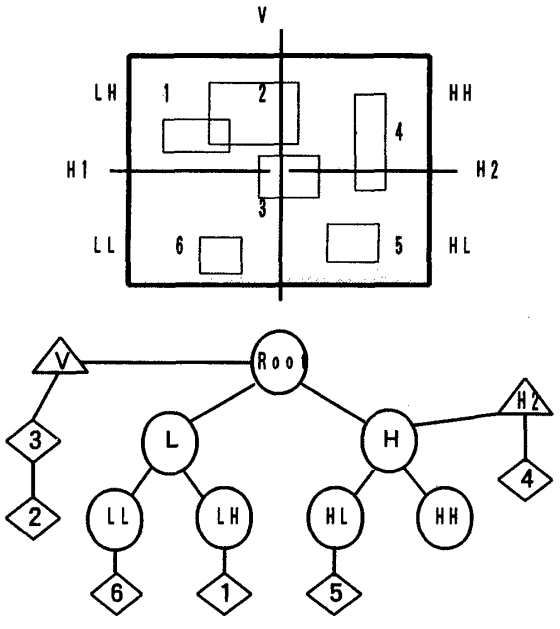

図１　Bucket Tree 構成方法

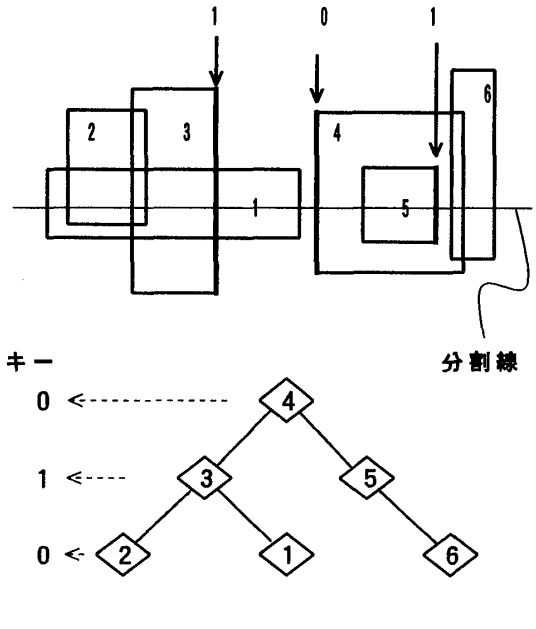

図２　2-d Bisector List

Bucket Tree Used as An Effective Data Structure for VLSI Layout System.
Hiroaki IWASAKI, Takashi MITSUHASHI
TOSHIBA Corp. ULSI Research Center,

3．評価方法

今回提案したBucket Tree との比較の対象として、従来用いられているQuad Tree [1] , K-d Tree [2] を採用し、表１に示したような実際のレイアウトデータを用いて、使用メモリ量，データ探索性能について調査した。

4．結果と考察

第３図に図形データ数に対する使用メモリ量を、第４図にデータＴにおける探索領域のサイズに対する探索ノード数をプロットしたものを示した。これらの図から、Bucket Tree はメモリ量がやや多いものの探索の性能においてはK-d Treeとほぼ同等であることがわかる。特に、レイアウト結果の修正というＧＥの正確を考慮した場合、特定の要素（セル・配線セグメントなど）に対する指示が頻繁に行われるため、小領域に対する探索に良好な性能を示しているBucket Tree は有効である。

今回の評価では、Tree分割の終了条件としてノード内の図形数で判断する場合と領域面積（ノードの深さ）で判断する場合の比較を行った結果、面積を基準とした手法が有効であることも確認された。

また、K-d Treeでは図形の位置の偏りによってバランスが極端に悪くなり、１点の探索時に10^3 程度の探索ノード数を要する場合も生じていたが、Bucket Tree ではチップ上の位置に影響されず良好な探索性能を示すことがわかった。

今後さらに検討を重ね、現在開発中のレイアウトシステムのデータベースとして実現していく予定である。

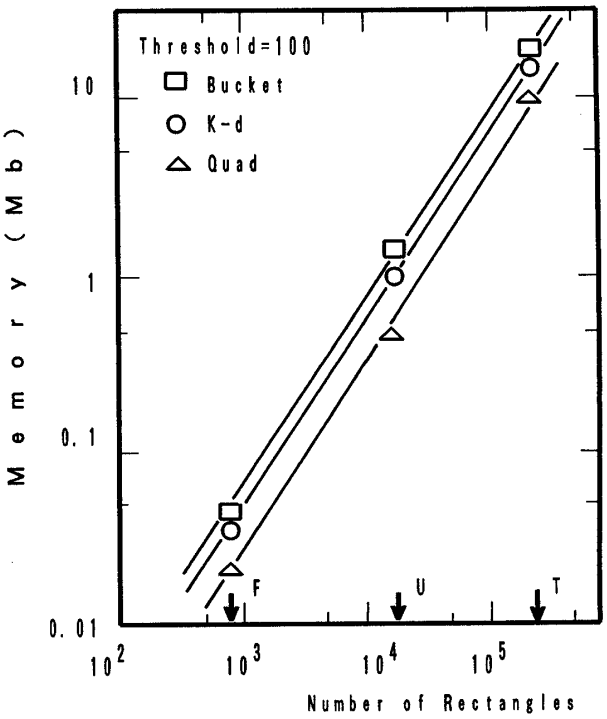

図３　図形データ数に対する使用メモリ量

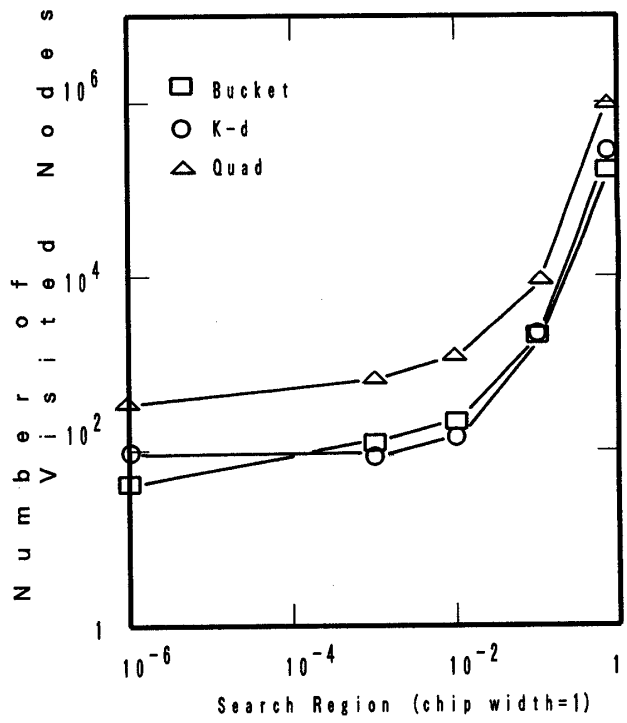

図４　探索領域サイズに対する探索ノード数

5．参考文献

1) G. Kedem, Porc. 19th DAC, p. 352, 1982.

2) J. B. Rosenberg, IEEE Trans. Computer-Aided Design, Vol. CAD-4, No. 1, p. 53, Jan 1985.

表１　データ仕様

Type	図形数	面積（μm^2）
F	693	1859.2 ×1890.0
U	18004	3892.0 ×3888.0
T	268109	9689.6 ×9690.0