設計自動化 72－2
(1994. 10. 27)

ハイパーグラフ分割のための動的クラスタリングに基づくヒューリスティックアルゴリズム

川本 貞行　若林 真一　小出 哲士　吉田 典可

広島大学 工学部
〒724 東広島市鏡山一丁目4番1号

あらまし

ハイパーグラフ分割問題はVLSIレイアウト設計等に応用される重要な問題であるが，NP-困難であることが知られている．そのため現在までにいくつかのヒューリスティックアルゴリズムが提案されているが，それらは大規模入力に対して高速に解を求めるものの局所解に陥り易かったり，逆に局所解には陥りにくいが大規模入力に対しては計算時間が著しく増大するものが多い．本研究ではこれらの問題に対処するために，節点のクラスタリングによって局所解を避けつつ質の良いヒューリスティック解を求めるハイパーグラフ分割ヒューリスティックアルゴリズムを提案し，実験的評価を行なう．

和文キーワード　ハイパーグラフ，グラフ分割，クラスタリング，ヒューリスティックアルゴリズム

A Heuristic Algorithm for Hypergraph Partitioning Based on Dynamic Clustering

Sadayuki KAWAMOTO　Shin'ichi WAKABAYASHI
Tetsushi KOIDE　Noriyoshi YOSHIDA

Faculty of Engineering, Hiroshima University
4-1, Kagamiyama 1-chome, Higashi-Hiroshima, 724 JAPAN

Abstract

The hypergraph partitioning problem is a very important problem and widely used in several applications, such as VLSI layout design. Since it is known to be NP-hard, several heuristic algorithms have been proposed. However, some of them tend to fall into a local optimum and others that can avoid falling into a local optimum need much computation time. To solve those drawbacks, we present a new heuristic algorithm for hypergraph partitioning based on dynamic clustering of hypergraph nodes, and show experimental results.

英文 key words　hypergraph, graph partitioning, clustering, heuristic algorithm

1 まえがき

論理回路は論理モジュールを節点に，ネットをハイパー枝に対応させることによりハイパーグラフでモデル化できる．論理回路分割問題とは論理回路が与えられたとき，論理モジュールをサイズの制約のもとにいくつかの集合に分割し，そのときの集合間にまたがるネット数(=カット数)の最小化を目的とする問題である．VLSIチップの設計においては，システム分割や論理モジュールの配置問題において論理回路の分割問題，すなわちハイパーグラフ分割問題は大変重要である[7]．しかし，近年のVLSIチップの高集積化により問題のサイズは大きくなってきており，また，ハイパーグラフ分割問題はNP困難な問題であるため効率の良いヒューリスティック手法が数多く提案されてきた[3] [5] [6] [10]．

中でもC.M.FiduccaとR.M.Mattheysesらによって提案された手法[3](以降FM法)はシンプルで高速であり，大規模な問題に対しても短時間に解を求めることができる．FM法は初期分割を行なった後，分割集合間で節点の移動を行なってカット数を減少させる反復改良法であるが，節点の移動が1個単位であることや分割集合サイズのバランスを崩すような移動が行なえないことにより節点移動を繰り返すうちにカット数の減少ができなくなり，最終的に局所解に陥りやすい欠点がある．

局所解を避けるには節点の移動を1個単位でなく数個単位で行なう手法が効果的である[2] [9]．さらに，枝によるつながりの強い節点同士を予めまとめて(クラスタ化して)おけば問題のサイズも小さくすることができるため，大規模な問題に対して有効であり，今までに多くのクラスタリングアルゴリズムが提案されてきた．[1] [4] [5] [11]．しかし従来のクラスタリング手法の多くではクラスタリングはあくまで分割の前処理として行われており，クラスタ構成要素は分割中は固定されているものが多い．この場合，問題サイズの減少と集合単位での節点移動は可能となるが一度局所解に陥ってしまうとそれからさらに解を改善することは困難である．しかしクラスタの構成を変化させれば新たな移動が可能となる場合があり，局所解からの脱出が実現できる．

クラスタを変化させながら分割を行なうアルゴリズムは過去に[2]で提案されている．このアルゴリズムでは階層的にボトムアップに構成されたクラスタを階層的にトップダウンに分解しながら節点の移動を行なっているが，分解が下位レベルにまで進んで局所解に陥ると，それ以上の改良が困難であると思われる．また，クラスタの分解だけでなく成長も行ないつつ節点移動を行なえばより広い解空間の探索が行なえ，良い解を得ることができると思われる．

そこで本稿では，分解と成長の操作を繰り返すことにより，クラスタを構成する節点集合を動的に変更しながらクラスタの移動による反復改良を行なうアルゴリズムを提案し，シミュレーション実験により有効性を示す．

2 準備

2.1 ハイパーグラフ

ハイパーグラフ $H = (V, E)$ は節点集合 $V = \{v_1, v_2, \ldots, v_n\}$ とハイパー枝集合 $E = \{e_1, e_2, \ldots, e_m\}$ からなり，各ハイパー枝は節点の部分集合で表される $(e_i = \{v_j, v_k, \ldots\})$ (図1参照)．もし，全てのハイパー

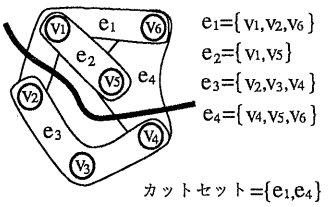


図1 ハイパーグラフ

枝 $e \in E$ に対し，$|e| = 2$ ならば H はグラフでもある．節点集合 V にその重みを表す関数が定義されていたならばハイパーグラフ $H = (V, E)$ は重みつきハイパーグラフであると言う．各節点の重みを v のサイズと言い $size(v)$ で表す．

重みつきハイパーグラフ $H = (V, E)$ に対し，自然数 k と関数 $P : V \to \{1, 2, \ldots, k\}$ が与えられたとする．このとき，$V_i = \{v | v \in V, P(v) = i\}(1 \leq i \leq k)$ で表される k 個の互いに素な節点部分集合を H の k 分割と定義する．そして各ハイパー枝 $e \in E$ に対し，スパン関数 $q(e) = \{P(v) | v \in e\}$ を定義し，$\{e | |q(e)| \geq 2, e \in E\}$ なるハイパー枝の集合を k 分割カットセットと言う．k 分割カットのサイズを k 分割カットセットに属しているハイパー枝の数(カット数)で表す．2分割の例を図1に示す．この例においてはカットセットは $\{e_1, e_4\}$，カット数は2となる．

2.2 問題の定式化

ハイパーグラフ k 分割問題とは，与えられたハイパーグラフをカット数が最小となるように k 個の節点集合に分割する問題である．

【ハイパーグラフ k 分割問題】

［入力］ ハイパーグラフ $H = (V, E)$
V: 節点集合, E:ハイパー枝集合, 整数 k
分割集合間のバランスの余裕 α, $(0 \leq \alpha \leq 0.1)$

［出力］ 目的関数を最小とする H の k 分割

［目的関数］ カット数

［制約条件］

$$max_{i,j \in \{1,\ldots,k\}} \{ | \sum_{u \in V_i} size(u) - \sum_{v \in V_j} size(v) | \} \leq max\{size(V)/k \times \alpha, max_{v \in V}\{size(v)\}\}$$

3 従来手法

3.1 節点の移動による反復改良アルゴリズム (FM 法)

グラフ分割問題に対する代表的なアルゴリズムであり，本稿で提案するアルゴリズムの一部としても使用する FM 法 [3] について,2分割の場合を例にとって簡単に説明する．

この手法では，まず各節点 v_i に対しゲイン $G(v_i)$ を定める．ゲイン $G(v_i)$ とは節点 v_i が現在の節点集合から別の集合に移動した時に減少するカット数である (図 2参照)．$G(v_i)$ を求めるためには,v_i につながる各ハイパー枝について，v_i の属している分割集合に v_i 以外の節点を持たない時+1, v_i の属している分割集合に全ての節点を持つ時−1 とし，v_i につながる全てのハイパー枝に対するこの値の和を求めれば良い．次に FM 法のアルゴリズムを述べる．

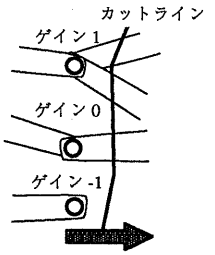


図 2 ゲインの求め方

STEP1:初期分割を行なう．

STEP2:全節点のゲインを求める．

STEP3:最大ゲインの節点を仮に移動し，移動先でロックする．ロックされた節点以外のゲインの更新を行ない，再び STEP3 を行なう．全ての節点がロックされるまでこの操作を繰り返すことによって移動した節点の系列 $N_1, N_2, \ldots, N_n$ とゲインの系列 $G_1, G_2, \ldots, G_n$ が求まる．

STEP4:求まったゲイン系列で，G_1 から加えていってその値が最大となる部分系列 $G_1, G_2, \ldots, G_i$ を求める．求めたゲインの部分系列に対応する節点系列 $N_1, N_2, \ldots, N_i$ の節点を実際に移動する．

STEP5:STEP4 まで求めた分割を初期分割として STEP2〜STEP4 を繰り返し，STEP4 で求めたゲインの和が 0 以下になると終了．

この手法では節点の移動は 1 個単位でしか行なわれず，また，カット数の増加するような移動を行なわないため図 3(a) のように局所解に陥ってしまうとこれ以上の節点の移動は行なえなくなり，最適解にたどり着けなくなる場合がある．もし，ここで図 3(b) の様に節点をいくつかのまとまり（クラスタ）にして移動すればさらにカット数を減らすことができる．

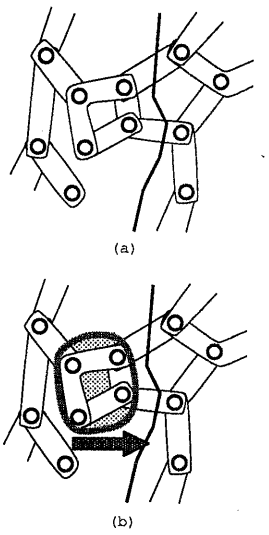


図 3 クラスタリングによるよる局所解の回避

節点のクラスラリングに関しては今までに様々な手法が提案されてきた [2] [4] [5] [9] [11]．その多くは与えられたグラフからつながりの強い節点集合からなるクラスタの集合を求めるだけのものであるが，このうち [2] [9] は節点移動による局所解を避けることを陽に考慮している．[9] ではカットライン付近の節点から深さ優先探索を行なうことによって移動に適した節点集合を見つけて移動を行なっている．この手法ではクラスタリングは移動対象となる節点集合に対してのみ適用されている．しかしこの手法では分割集合サイズのバランスが崩れやすく，最終的にバランスした解を求めるために解の質が落ちてしまう場合がある．[2] では予めボトムアップにクラスタを構成して行き，クラスタが 2 個になるまでマージを行なう．次に移動を行ないながらトップダウンにクラスタを分解して行く．この手法では初期段階ではサイズの大きな集合単位で移動を行なうが移動単位は次第に小さなサイズとなり，最終的には 1 節点単位での移動が行なえる．この様に様々な集合単位での移動により局所解に陥りにくくしている．しかし，この手法ではクラスタ集合は分解操作だけが行なわれており，もしここで分解だけでなくクラスタの成長も行ないながら節点移動による改良を行なえば，さらに探索空間を広げることができ，より良い解を求めることが可能になると考えられる．

本稿で提案するアルゴリズムでは，クラスタ集合に対し分解と成長の操作を適用しながらクラスタ移動を繰り返し，局所解を避けることによってさらなる改良を試みるものである．

4 アルゴリズム

4.1 提案アルゴリズムの概要

提案アルゴリズムについて説明する．提案アルゴリズムは大きく分けてクラスタ変更フェーズとクラスタ移動による分割改良フェーズからなっている．アルゴリズムのフローチャートを図4に示す．

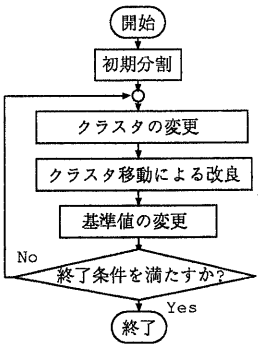


図 4 フローチャート

図5に提案アルゴリズムの実行例の一部を示した．

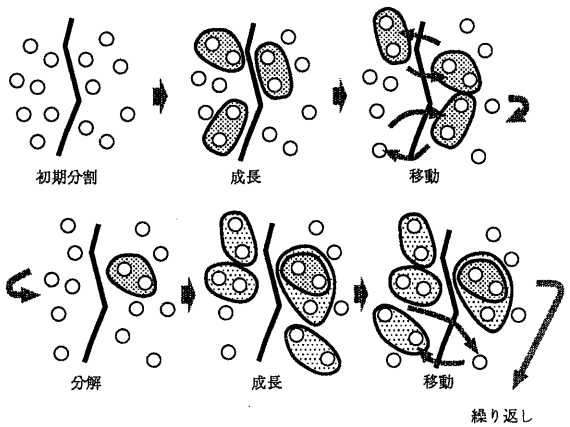


図 5 提案アルゴリズムの動き

入力ハイパーグラフは最初に初期分割が行なわれる．クラスタ変更フェーズでは，クラスタの分解，成長を行ない，クラスタ集合を変化させる．分割改良フェーズでは，変更後のクラスタ集合に対してクラスタ移動による分割改良 (FM 法) を適用し，カット数の減少を試みる．以上の2フェーズを1サイクルとし，繰り返し改良を行なう．

以降，提案アルゴリズムの詳細について説明を行なって行く．始めに初期分割について，次にクラスタ変更について説明を行ない．最後にクラスタ移動による分割改良について説明する．

4.2 初期分割

初期分割ではランダムに節点を k 個の部分集合に分割する．提案アルゴリズムでは節点をクラスタ単位で扱うため，バランスに関する制約条件を満たしながら繰り返し改良を行なうことが困難なため，初期にはバランス制約を緩めておき，繰り返しが進むにつれてバランス制約を厳しくし，最終的には課せられた制約を満たす解を得るようにしている．具体的にはバランス制約に関するパラメータ αを，初期には$\alpha \times \beta$としてβ倍の余裕を与え，このβの値を繰り返し毎に減らして最終的には$\beta = 1$となるようにしている．したがって初期分割は制約条件を満たさない場合もある．

4.3 クラスタ変更フェーズ

クラスタ変更フェーズではクラスタの分解と成長を行なう．後の移動による分割改良でカット数の少ない分割を行なうことを容易にするためにクラスタに求められる条件は，枝による接続関係の強い節点同士が同じクラスタに属すること，及び，他のクラスタとつながる枝が少ないこと，である．クラスタ変更フェーズでは，条件に適したクラスタをつくり出すことを目的として，まず，ある基準によってクラスタの評価を行ない，基準を満たさないクラスタは分解する．次に，クラスタ同士のマージによって成長を行なう．以下ではクラスタの分解と成長の操作について説明を行なう．

4.3.1 クラスタ分解

各クラスタ c_iは以下の式で表される評価値を持つ．クラスタ内枝数とは，そのクラスタ外の節点にはつながっていないクラスタ内のハイパー枝の数のことである．これにより，より枝の密度が高い節点集合が大きな評価値を持つこととなる．

$$eval(c_i) = \frac{\text{クラスタ内枝数}}{\text{クラスタ内節点数}}$$

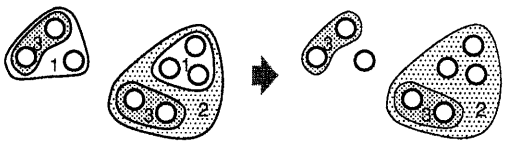
アルゴリズム中では次式のような関数によって表される評価基準値があり，評価基準値を満たさないクラスタを分解することで質の良いクラスタを残して行く．

$$\text{基準値} = (\text{評価値の平均}) \times \frac{\text{全節点数}}{\text{現在のクラスタ数}} \times\{1+\sin(\frac{\pi}{4}\cdot(\text{改良が行なわれなかった回数}))\}$$

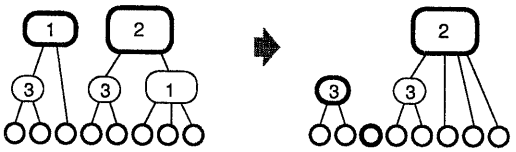
ここで評価値の平均とはこの時点での全クラスタの評価値の平均値のことであり，改良が行なわれなかった回数とはアルゴリズムの繰り返し実行において，移動による分割改良によっても現在までに連続してカット数の改善が行なわれなかった回数である．この関数では (全節点数)/(現在のクラスタ数) によってクラスタ数が多い時には基準値は減少して成長によるクラスタ数を減少を促し，

クラスタ数が少ない時には基準値は増加して分解によるクラスタ数の増加を促す．また，数回にわたって移動による分割改良が行なわれない場合は正弦関数によって基準値を大きく変動させることによりクラスタを大きく変化させて局所解からの脱出を図る．

クラスタは図 6(a) のように階層的に構成されている．これを実現するために，データ構造は図 6(b) の様に各クラスタは木構造となっており，全ての階層レベルにおいて構成要素と評価値を保存している．分解手続きは階層中の全ての評価値を満たさないクラスタに対して適用する．この全階層の分解操作によって，より変化に富んだクラスタ集合をつくり出す．



(a)



(b)

図 6 クラスタの分解

4.3.2 クラスタ成長

基準値を満たさないクラスタの分解が行なわれた後，各クラスタは他のクラスタとマージを行なうことによって成長する．2つのクラスタの結合によって生じる新たなクラスタの外部につながる枝ができるだけ少なくなるようにマージを行なう．まずマージするクラスタ対を接続度を基準にして求めて，接続度の大きなものからマージを行なっていく [2]．以下にマージによるクラスタ成長の手続きについて述べる．

N_cを2つのクラスタが共通に持つハイパー枝集合とし，クラスタリングにおける任意の2クラスタ c_1, c_2の接続度 $F(c_1, c_2)$ を次のように表す．

$$F(c_1, c_2) = \sum_{w \in N_c} (f(n_w) - f(n_w - 1))$$

ここで n_wはハイパー枝 wが接続するクラスタ数，$f(n_w)$ は n_wに関する評価値で図 7のような関数である．2クラスタからなるハイパー枝に接続するクラスタを1つのクラスタにマージした場合が最も目的関数の減少度が大き

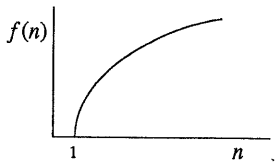


図 7 $f(n)$

く，多くのクラスタを持つ枝につながるクラスタをマージするほど目的関数の減少度は低くなる．

以下にマージによるクラスタ成長アルゴリズムを示す．

【クラスタ成長アルゴリズム】

STEP1: 分解操作終了後のクラスタの集合を Cとする．

STEP2: C内の各クラスタに対し，接続度が最も大きい相手のクラスタを求める．その時の接続度をクラスタの結合値と呼ぶ．

STEP3: $C = \phi$まで STEP3a,3b,3c を繰り返す．

STEP3a: 最大の結合値を持つクラスタ c_{max}と，STEP2 において計算した相手のクラスタ c_cとをマージする．

STEP3b: Cから c_{max}と c_cを削除する．

STEP3c: c_{max}と c_cに接続を持つクラスタの結合値を更新する．

このアルゴリズムでは全てのクラスタペアに対して接続度を覚えておく必要はなく，各クラスタは隣接するクラスタ中で最も接続度の大きな相手との結合値を覚えておけば良い．また結合値の更新も c_{max}と c_cに隣接するクラスタだけ行なえば良く，使用メモリ，計算時間とも節約が図れる．

4.4 クラスタ移動による分割改良

クラスタの移動による反復改良は2分割の場合は FM 法，k分割の場合は FM 法を拡張した文献 [8] の手法を用いた．

アルゴリズムの終了条件は，規定回数以上繰り返し改良が行なわれず，かつバランス制約を満たしている時に満たされるものとする．

5 実験

5.1 実験方法

提案手法を Sun microsystems 社の SPARC Server 1000 上で C 言語を用いて実現し，シミュレーション実験を行なった．実験データは表 1 の MCNC ベンチマークデータを用いた．これらは実際の論理回路に基づいたデータである．

表 1.MCNC ベンチマークデータ

データ名	fract	prim1	ind1	prim2	ind2
節点数	149	914	2851	3132	12637
枝数	147	985	2478	3136	13419

分割数は2とし，比較手法として通常のFM法を採用した．バランス制約におけるαの値は0.01とした．すなわち均等に2分割した場合の各部分集合サイズの1%のサイズ差を許している．提案手法では初期には50%の差を許し，アルゴリズムの繰り返し1回毎に1%づつ制約を厳しくし，最終的に制約を満たすようにしている．また，提案手法の終了条件は10回連続して分割改良が見られなかった時に満たされることとした．

提案手法，比較手法とも最終的な解が初期分割に依存するため10個の異った初期分割 (ind2は時間の都合上5個) に対して実験を行ない，その平均で比較を行なった．

5.2 実験結果と考察

表2に比較手法の実験結果を，表3に提案手法の実験結果を示す．ave.CUTは平均カット数，min.CUTは求めた最少カット数，ave.TIMEは平均計算時間 (単位は秒) を表す．

表2.FM法の実験結果

データ名	ave.CUT	min.CUT	ave.TIME
fract	21.6	11	0.3
prim1	82.8	64	5.2
ind1	57.1	36	236.0
prim2	313.9	239	36.8
ind2	935.0	817	6090.1

表3. 提案手法の実験結果

データ名	ave.CUT	min.CUT	ave.TIME
fract	11.0	11	3.3
prim1	56.3	45	44.9
ind1	66.3	26	1378.0
prim2	184.3	152	389.0
ind2	496.8	305	23693.1

実験結果より提案手法はind1を除いた全てのデータに対して比較手法より少ないカット数の解を求めていることが分かり，動的クラスタリングの有効性が確認できる．

カット数による比較では，提案手法は比較手法に対し平均で37.1%カット数が減少している．計算時間は提案手法は比較手法に比べ平均8.0倍増加しているが，提案手法は繰り返しの回数が比較手法に比べて多い (平均5.1倍) ことも考えると比較的高速で実用的な計算時間であると言える．

ind1については最少カット数は比較手法より良い解を求めているものの平均カット数の比較では負けている．これはおそらく実験データの特性によるもので，クラスタリングの効果を得にくいデータ，つまり節点次数の平均が小さく，均一に枝の分散したグラフで，特に結合の強い節点集合が少ないためだと思われる．詳細は今後，データの解析と実験によって解明する予定である．

6 あとがき

本稿ではハイパーグラフ分割問題に対し，節点集合の動的なクラスタリングによって局所解の回避を試みるヒューリスティックアルゴリズムを提案し，実験的評価を行なった．実験結果より提案手法はクラスタリングを行なわない従来手法より良い解を比較的少ない計算時間で求めることが分かった．今後の課題として，実験に基づいた基準値関数，及びその他のパラメータのより良い設定，FM法以外の他の従来手法との比較，分割改良フェーズに用いるFM法に代わる新たなアルゴリズムの開発，が挙げられる．

謝辞

実験を行なうにあたって御協力いただいた本学学部生 小林 正英君に感謝します．なお，本研究の成果の一部は文部省科学研究費補助金試験研究 (B)(2)(課題番号 06558042) によるものである．

参考文献

[1] J. Cong and M. Smith: "A parallel bottom-up clustering algorithm with application to circuit partitioning in VLSI design," Proc. of 30th Design Automation Conference, pp. 755–760 (1993).

[2] 枝廣, 吉村："階層クラスタリング法を用いたセル列型LSIのための配置手法," 信学技報, VLD 90-62 (1990).

[3] C. M. Fiduccia and R. M. Mattheyses: "A linear-time heuristic for improving network partitions," Proc. of 19th Design Automation Conference, pp. 175–181 (1982).

[4] J. Garbers, H. J. Promel and A. Steger: "Finding clusters in VLSI circuits," Proc. of International Conference on Computer-Aided Design, pp. 520–523 (1990).

[5] L. Hagen and A. B. Kahng: "New spectral methods for ratio cut partitioning and clustering," IEEE Trans. Comput.-Aided Design of Integrated Circuits & Syst., Vol. CAD-11, No. 8, pp. 1044–1051 (1992).

[6] A. B. Kahng: "Fast hypergraph partition," Proc. of 26th Design Automation Conference, pp. 762–766 (1989).

[7] 上土井, 川本, 若林, 小出, 吉田："ハイパーグラフ k分割手法に基づくスタンダードセル配置手法," 第47回 (平成5年後期) 情報処理学会全国大会講演論文集, Vol. 6, pp. 109–110 (1993).

[8] L.A.Sanchis: "Multiple-way network partitioning," IEEE Trans. on Compt., Vol. 38, No. 1, pp. 62–81 (1989).

[9] Y. Mimasa: "Graph partitioning algorithms using subgraph migration for VLSI design," Master thesis, Graduate School of Engr. Hiroshima Univ. (1994).

[10] Y. C. Wei and C. K. Cheng: "Towards efficient hierarchical designs by ratio cut partitioning," Proc. of International Conference on Computer-Aided Design, pp. 298–301 (1989).

[11] Y. C. Wei and C. K. Cheng: "A two-level two-way partitioning algorithm," Proc. of International Conference on Computer-Aided Design, pp. 516–519 (1990).