節点集合の移動に基づくグラフ分割手法

3T－2

三政 義康† 礒本 和典‡ 若林 真一† 小出 哲士† 吉田 典可†

†広島大学工学部　‡マツダ株式会社

1 まえがき

本稿ではVLSIレイアウト設計における配置設計や分散処理におけるタスク割り当て等において重要な問題の１つであるグラフの分割問題について考察する．この問題に対する従来法としてはFM法[1]，KL法[2]等が知られているが，これらの手法は１個の節点の移動，交換に基づいているため局所解に陥りやすいという問題点がある．最近では，節点を集合単位で移動するアルゴリズムが幾つか提案されている[3][4]．そこで本稿では，グラフの節点部分集合の移動に基づくヒューリスティックアルゴリズムを提案し，その実験的評価を行う．

2 準備

グラフ２分割問題を以下に示す．

【グラフ２分割問題】

[入力] 無向グラフ　G=(V,E).

[出力] 条件を満足し，目的関数を最小とする節点素な集合V_1,V_2へのVの分割．

[制約条件] $||V_1|-|V_2||\leqq 1$，$V_1 \cup V_2=V$，$V_1 \cap V_2=\phi$．

[目的関数] $Z=|C|$，ここで$C=\{(v',v'') \mid (v',v'')\in E, v'\in V_1, v''\in V_2\}$．以下では，Cをカット，$Z=|C|$をカット数と言う．

□

次に用語の定義を行う．

境界節点：$(v_i,v_j)\in C$のとき，節点v_i,v_jを境界節点といい，v_{bi},v_{bj}で表す．境界節点の集合をBとする．

節点部分集合SG_i：$SG_i \subset V_i (i=1,2)$.

外部接続枝：$ex\_edge(SG_i)=\{(v_k,v_l) \mid v_k\in SG_i, v_l\in V_{3-i} (i=1,2)\}$.

内部接続枝：$in\_edge(SG_i)=\{(v_k,v_l) \mid v_k\in SG_i, v_l\in V_i (i=1,2)\}$.

3 提案アルゴリズム

3．1 従来手法

グラフ２分割問題はＮＰ-完全のクラスに属することが知られている[3]．そのため，これ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．代表的な手法であるＦＭ法（ＫＬ法）は，

Step1：初期分割．

Step2：移動（交換）することでカット数の減る節点を１つ（ペアを）見つけて移動（交換）する．

Step3：そのような節点がなくなったら終了．

という手法であり，１個の節点の移動，交換を基本としているので局所解に陥りやすいとという問題点がある[3]．

提案アルゴリズムは任意に与えられる初期分割を逐次的に改良する繰り返し改良法であるが，従来手法と異なるのは移動対象を節点部分集合にしたところにある．節点１個単位の移動では図１のv_1節点を移動した結果得られるカット数は増加し，移動の対象とはなりにくいが，節点集合$\{v_1,v_2,v_3,v_4\}$を移動した場合，カット数は減少し，従来手法では移動の対象となりにくかった節点の移動を行うことができる

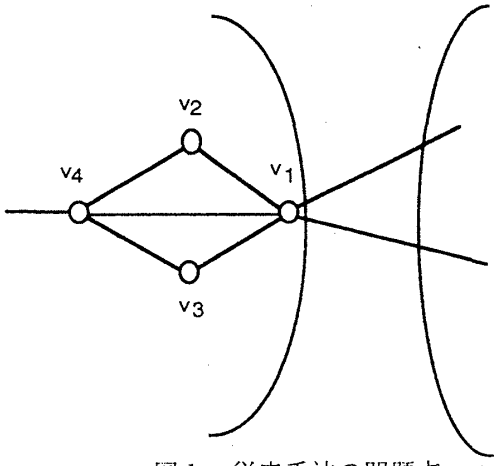

図１　従来手法の問題点

A graph bisection algorithm based on subgraph migration,Yoshiyasu MIMASA†, Kazunori ISOMOTO‡, Shin'ichi WAKABAYASHI†,Tetsushi KOIDE†, Noriyoshi YOSHIDA†, † Faculty of Engineering, Hiroshima University. ‡Technical Research Center, MAZDA Motor Corporation.

ので，より広い解空間の探索を行うことができる．特に局所的に強い結合がいくつか存在するグラフに対しては良い解を求めることが期待できる．

3．2　アルゴリズムの概要

本アルゴリズムでは図2に示すように|ex_edge(SG)|＞|in_edge(SG)|となるような節点部分集合SGを境界節点から求め，その節点集合の移動を繰り返すことで局所解に陥らないようにしている．

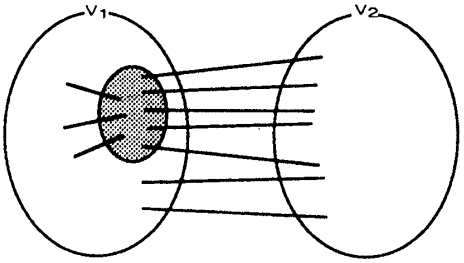

図2　節点部分集合

以下にアルゴリズムＡＧＢ（Algorithm for Graph Bisection）の概要を示す．

【アルゴリズムＡＧＢ】

Step1：初期分割を行い，$P=V_1$，$Q=V_2$とする．

Step2：PにおいてM個の境界節点を選び，境界節点から深さkまでの幅優先探索を行う（kは正の定数）．

Step3：Step2において複数の境界節点から探索された各節点について，それぞれの境界節点まで深さkの後進探索を行い，境界節点に達したすべての経路上の節点を節点部分集合SGにいれる．

Step4：｜ex_edge(SG)｜＞｜in_edge(SG)｜ならばSGをQに移動，そうでなければStep2に戻る．

Step5：$|V_1|<|V_2|$ならば$P=V_2$，$Q=V_1$とし，そうでなければ$P=V_1$，$Q=V_2$として，Step2からStep4を繰り返す．

Step6：$|V_1|<|V_2|$ならば$P=V_2$，$Q=V_1$とし，そうでなければ$P=V_1$，$Q=V_2$として，Mの値を減らしてStep2からStep5を繰り返す．

Step7：1個単位で節点の移動を行い節点数を均等にする．

3．3　アルゴリズム動作例

アルゴリズムＡＧＢの動作例を図3に示す．

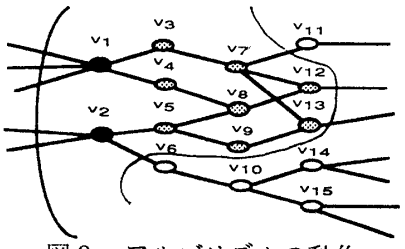

図3　アルゴリズムの動作

図3において，節点v_1と節点v_2は境界節点である．kを3とするとStep2において，v_1から開始した幅優先探索（ＢＦＳ）によって探索された節点は$\{v_3,v_4,v_7,v_8,v_5,v_{11},v_{12},v_{13}\}$となり，$v_2$から開始した幅優先探索によって探索された節点は$\{v_5,v_6,v_8,v_9,v_{10},v_4,v_{12},v_{13},v_{14},v_{15}\}$となる．ここでStep3において，節点$v_8,v_{12},v_{13}$は2つの境界節点からともに探索されているので，この節点から後進探索を行い，節点$\{v_1,v_2,v_3,v_4,v_5,v_7,v_8,v_9,v_{12},v_{13}\}$が節点部分集合SGとなる．Step4においてSGは|ex_edge(SG)|=5，|in_edge(SG)|=4なので，これをQに移動する．ここで，$|V_1|<|V_2|$ならば，PとQを入れ替えて，Pについて再び，上記の処理を行う．

4　あとがき

グラフの2分割問題に対し，アルゴリズムを提案した．今後の課題としてはアルゴリズムを計算機上に実現し，従来手法であるＫＬ法との比較がある．また，アルゴリズムの並列化，VLSI配置アルゴリズムへの適用を考えている．

参考文献

[1] C.M.Fiduccia and R.M.Matteyses : "A linear-time heuristic for improving network partitions," Proc.19th DAC.,pp.175-181 (1982).

[2] B.W.Kernighan and S.Lin : "An efficient heuristic procedure for partitioning graphs," Bell Sys. Tech. J. , No.49,pp.291-307 (1970).

[3] 黒岩，他: "並列計算機Cenju2上での並列グラフ分割アルゴリズムの実装と評価,"第6回軽井沢ワークショップ論文集,pp.543-548 (1993).

[4] J.E.Savage and M.G.Wlka : "Parallelism in graph-partitioning," Journal of Parallel and Distributed Computing,No.13,pp.257-272 (1991).