計測自動制御学会東北支部 第229回研究集会 (2006.6.9)
資料番号 229-7

貪欲算法に基づくロジックインメモリVLSIのハイレベルシンセシス

High Level Synthesis of a Logic-In-Memory VLSI Based on a Greedy algorithm

○櫻田　卓也*，工藤　隆男*，亀山　充隆**

○Takuya Sakurada*, Takao Kudoh*, Michitaka Kameyama**

*八戸工業高等専門学校, **東北大学大学院情報科学研究科

*Hachinohe National college of Technology, **Graduate School of Information Sciences, Tohoku University

キーワード： VLSIプロセッサ（VLSI Processor），ハイレベルシンセシス(High-Level-Synthesis)，ロジックインメモリ構造(Logic-in-Memory Architecture)，アロケーション・スケジューリング(Allocation / Scheduling)，貪欲算法(Greedy Algorithm)

連絡先： 〒039–1192　青森県八戸市田面木字上野平16–1 八戸工業高等専門学校 電気情報工学科
工藤隆男，Tel.: (0178)27-7279，Fax.: (0178)27-7279，E-mail: tkudoh-e@hachinohe-ct.ac.jp

1. はじめに

知能アルゴリズムと集積回路技術を融合したリアルワールド応用知能集積システムの実現には，動的に変化する環境への高速応答性が重要になる．[1, 2, 3]

また，VLSI製造技術の進展に伴う高集積化のため，ハイレベルシンセシスの重要性が高まっている[4]．

処理すべきアルゴリズムがデータ依存グラフ(DFG)で与えられるとき，DFGに条件分岐がない処理を最高性能で実現するVLSIの設計においては，演算ノードの開始ステップの決定（スケジューリング）と演算ノードを実行する演算器の決定（アロケーション）を，可能な組合せの中からあらかじめ探索しておくことが可能である．条件分岐のないDFGで表されるアルゴリズムには，高速フーリェ変換や絶対差分演算，行列演算，相関演算など知能集積システム実現のために必須の演算が挙げられる．

しかしながら，外界からの信号に応じ処理を瞬時に切り替えながらの高速応答性を必要とする演算のように，処理アルゴリズムが条件分岐を有するDFGで表される場合，最適なアロケーションとスケジューリングは条件により異なることから，あらかじめ決定することは困難である．よって，条件分岐後の処理に必要なアロケーションとスケジューリングをその都度高速に行うリアルタイムハイレベルシンセシスが重要になる．

また，ディープサブミクロンVLSIプロセッサにおいては，シグナルインテグリティや配線遅延などの配線に起因する性能劣化が深刻になっている[5]．特に，メモリ部と演算部との間に複雑な相互結

合回路網を備えるアーキテクチャにおいては，相互結合回路網が大規模になればなるほど配線問題が深刻になりがちである．このことから，シンプルな相互結合回路網を有する並列構造VLSIアーキテクチャが重要になる．

本稿では，そのようなハードウェアモデルとして，演算器に専用のローカルメモリを備えるモジュールを，シンプルな相互結合回路網で結合するロジックインメモリアーキテクチャを対象とし[6, 7, 8, 9]，演算器数制約下で処理時間最小化を達成するリアルタイムハイレベルシンセシスの手法を提案する．対象とするハードウェアモデルの場合，アロケーションとスケジューリングは互いに依存することから，最適解の探索問題は組み合わせ問題になる．厳密な意味での最適解を探索するためには，総当り的探索を行えばよい．しかしながら大規模なDFGの場合，組合せが爆発することから，高速応答性は期待できない．

このような問題を解決する方法として，演算ノードのスケジューリングの自由度（モビリティ）を用い，クリティカルパス上の演算を特定の演算器にアロケーションする．次に，モビリティが小さいノードから，順に，局所的な転送時間が最小になるアロケーションとスケジューリングを行う方法を提案する．

この方法は，クリティカルパス上の演算を特定の演算器にアロケーションすることにより，演算器間のデータ転送時間を不要とし，さらには，クリティカルパス以外のノードについては，局所的なデータ転送時間が最小になるようにアロケーションとスケジューリングを行うことにより，最適解の探索を行おうとする方法である．

最後に，提案の方法は，与えられる処理アルゴリズムに対し，処理時間を最小化するアロケーションとスケジューリングを高速で行えることを，実例を通して明らかにしている．

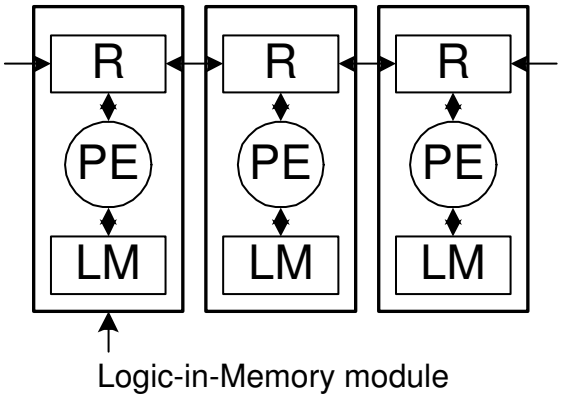


Fig. 1　ハードウェアモデル

2.　ハードウェアモデル

データ転送における配線遅延を小さくするためには，短い配線によるデータ転送を可能とする規則的な構造のハードウェアモデルが望まれる．このようなハードウェアモデルとしてFig. 1 のように1個の処理要素(PE)と1ワードのレジスタ(R)とローカルメモリ(LM)を一体化したロジックインメモリモジュールからなる空間並列構造のハードウェアモデルをとりあげる．

モジュール間のデータ転送をレジスタを介して隣接するモジュールに限定することにより，データ転送用の配線を短くできる．各PEはDFGの処理に必要な演算器，たとえば加算器や乗算器などの演算器を備えており，アロケーションを行う演算ノードの種類に応じて演算を切り替えて実行できるものとする．

このハードウェアモデルの利点は，負荷分散タイプやデータ再利用タイプなどのように演算ノードの転送が少ないか，規則性のある転送のクラスに適する．

以下の例については，隣接するモジュール間のデータ転送に要する時間は1クロックステップとし，演算に要する時間は1クロックステップとする．

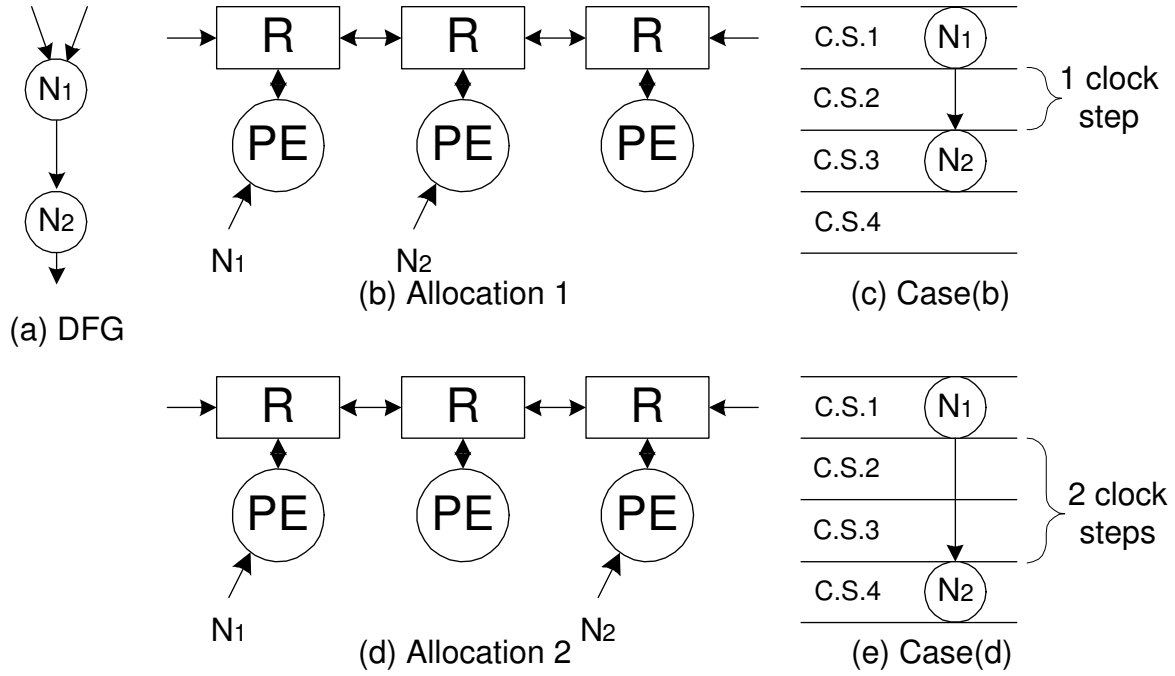


Fig. 3　アロケーションとスケジューリングの依存性

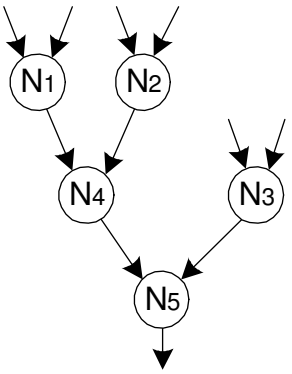


Fig. 2　データ依存グラフ (DFG)

3. アロケーションとスケジューリングの統合の必要性

PE数を限定する本ハードウェアモデルに基づくプロセッサ上で，DFGで与えられる処理アルゴリズムを最小処理時間で実行する場合，データ転送時間と演算時間とを合わせた処理時間全体の最小化が重要になる．Fig. 2のDFGは，演算の依存関係を表す．ノードは加算や乗算等の演算を表す．矢印はデータの依存関係を表す．矢印の根本のノードは親ノード，矢印の先のノードは子ノードと呼ぶ．子ノードの演算は親ノードの演算結果を必要とする．

Fig. 3に示すようにDFGのノードをPEにアロケーションする場合，PE間のデータ転送に必要なクロックステップ数は，転送のために通過するモジュール数に依存するため，スケジューリングはアロケーションが確定しないと決定できない．(a)のDFGに対して(b)のようにアロケーションする場合，PE_1とPE_2は隣接しているので，通信時間は1クロックステップ必要である．一方，(d)のようにアロケーションする場合，通信時間は2クロックステップ必要である．

このようにアロケーションとスケジューリングは不可分の関係にあることから，アロケーションとスケジューリングを統合する設計法が必要になる．

アロケーションとスケジューリングを同時に行うため，Fig. 4のように横方向にPE番号，縦方向にクロックステップをとった表を用いる．この表

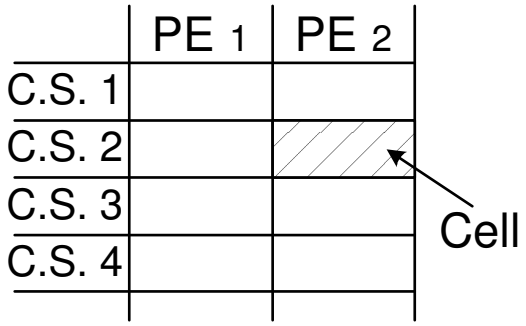


Fig. 4　アロケーションとスケジューリングの探索空間

にノード番号を書き込み配置していくことによりアロケーションとスケジューリングを同時に行う．以降は，この表中の一つの升目をセルと呼ぶことにする．

4.　提案法を用いた探索

4.1　問題の枠組み

Fig. 5のように条件分岐を有するDFGの場合，セクションCの最適なアロケーションとスケジューリングは，2通り存在する．もし，N通りの経路が存在する場合，N通りの最適解が存在することから，後ろのセクションになるほど，Nが膨大になり，N通りの最適解をあらかじめ記憶しておくためのメモリ容量が大きすぎるという問題がある．このような問題に対して，条件に応じて，セクションごとの最適解をリアルタイムで探索することが重要になる．

最適解の探索問題はFig. 4のセルへのノード配置問題となる．全ての可能な組み合わせを探索すれば，最適解を探索できる．しかし，総当たり的探索では，ノード数が多い場合，組み合わせ数が爆発することから解の高速探索は困難になる．そこで，近似解の探索が重要になる．

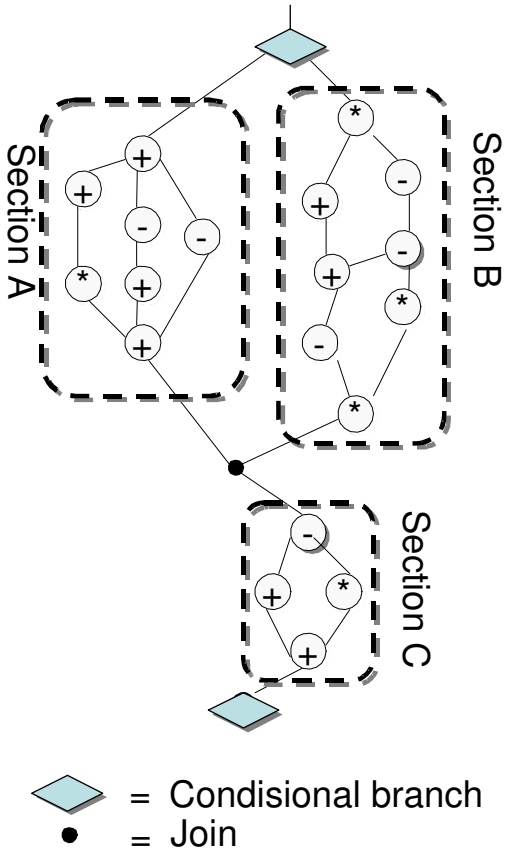


Fig. 5　条件分岐を有するDFG

4.2　貪欲算法

貪欲算法は，優先度に基づき，局所的な最適解を成長させることにより，最終的に近似解を導き出す方法である．局所解に陥りやすいという欠点はあるものの，高速探索を可能とするという特徴に着目し，リアルタイムハイレベルシンセシスへの適用を考える．

貪欲算法を用いると，Fig. 4のセルに全くノードが配置されていない状態から始まり，優先度に従って逐次的にノードを配置することにより解を構成できる．セルへの配置は，処理時間を可能な限り最小化するため，親ノードから子ノードへの局所的な転送時間を最小にするセルへの配置を行う．すなわち，クロックステップ差が小さいセルに配置する．配置するセルの候補が複数個ある場合にはPE番号の小さいセルに配置する．

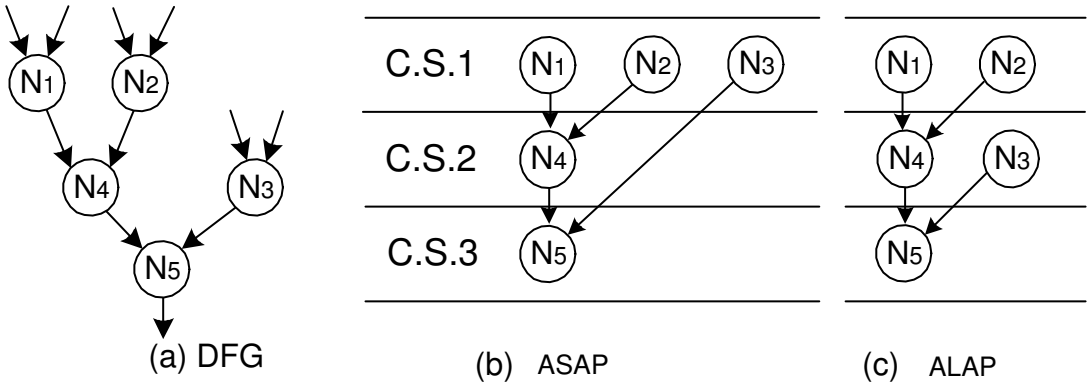


Fig. 6　ステップ１

4.3　モビリティ

配置可能なセルが少ないノードから配置するため，優先度には，モビリティを用いる．モビリティはASAPスケジューリングとALAPスケジューリングから求める．２種類のスケジューリングのクロックステップの差が，各ノードのモビリティとなる．

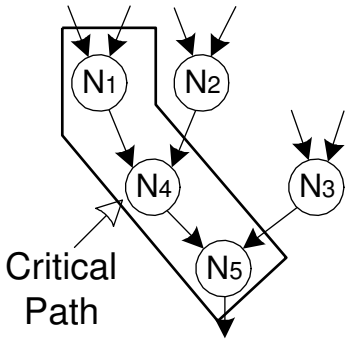


Fig. 7　ステップ２

4.4　解の探索

解の探索手順について述べる．以下の説明では，モビリティが0のノードを連ねた一つの経路をクリティカルパスと呼ぶ．親ノードから子ノードへのデータ転送に要する時間を局所転送時間と呼ぶことにする．

解の探索について述べる．まず，局所転送時間を最小化するため，クリティカルパスのノードを同一のPEで連続したクロックステップに配置する．これにより，局所転送時間を0にできる．

次に，モビリティの小さいノード（但し，同じモビリティのノードが複数あるならノード番号の大きいノード）から順にセルへ配置していく．このときの配置するセルを選択する．その基準は局所転送時間を最小にするセルに配置するものとする．

これにより局所的な転送時間を最小化しつつ，スケジューリングとアロケーションを同時に行える．

4.4.1　ステップ１：モビリティの算出

Fig. 6に示すように，モビリティはASAPスケジューリングとALAPスケジューリングから求める．例として，(a)のDFGのモビリティを求める．(b)はASAPスケジューリングで，(c)はALAPスケジューリングである．ASAPスケジューリングは，PEの数を無制限とし，また，転送時間を考慮しない場合に，それぞれのノードを可能な限り早いクロックステップにスケジューリングしたものであり，同様に，ALAPは最も遅いクロックステップにスケジューリングしたものである．この2種類のスケジューリングでのクロックステップの差が，それぞれのノードのモビリティとなる．N_3のモビリティは1，その他のノードのモビリティは0である．

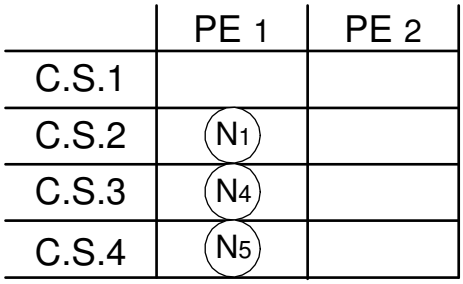


Fig. 8　ステップ3

4.4.2　ステップ2：クリティカルパスの選択

クリティカルパスを求めるために，モビリティが0のノードを連ねた一つの経路を選び出す．複数の候補がある場合，経路の先頭のノード番号が小さいものを選び出す．Fig. 7では，クリティカルパスの候補として，N_1・N_4・N_5の経路とN_2・N_4・N_5の経路があるが，N_1・N_4・N_5の経路をクリティカルパスとしている．

4.4.3　ステップ3：クリティカルパスのノードの配置

クリティカルパス上のノードの局所転送時間を0にするため，Fig. 8のように，クリティカルパス上のノードを同じPE番号の連続したクロックステップへ配置する．

4.4.4　ステップ4：それ以外のノードの配置

クリティカルパス上に存在しないノードは，モビリティの小さい順に配置する．モビリティが同じノードが複数ある場合は，ノード番号が大きいノードから配置することとする．

配置するセルの選択は，子ノードとの依存関係を満たすセルのうち一番大きなクロックステップのセル，すなわち，局所的転送時間が最小になるセルを選択する．この例ではN_2の配置となる．N_2を配置可能なセルはFig. 9(a)の網掛けのセルである．網掛けのセルのうち，PE番号の小さいセルへ配置する．その後にN_3を配置する．Fig. 9(b)の網掛けのセルに配置可能である．配置可能なセルのうちクロックステップが最大のセル，すなわち局所転送時間が最小となるセルへ配置する．解をFig. 9(b)に示す．

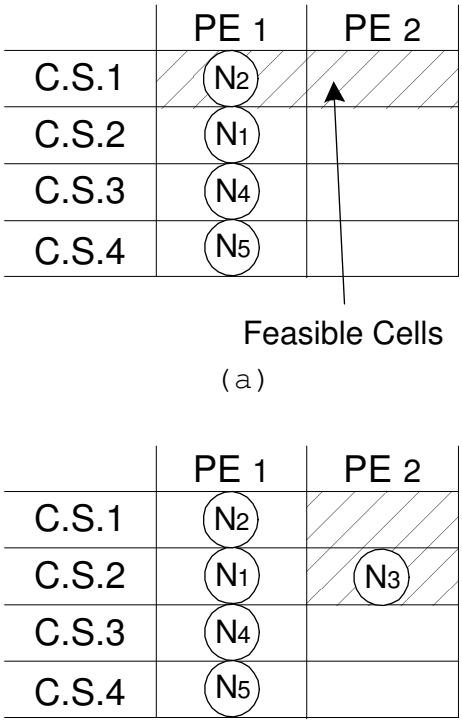


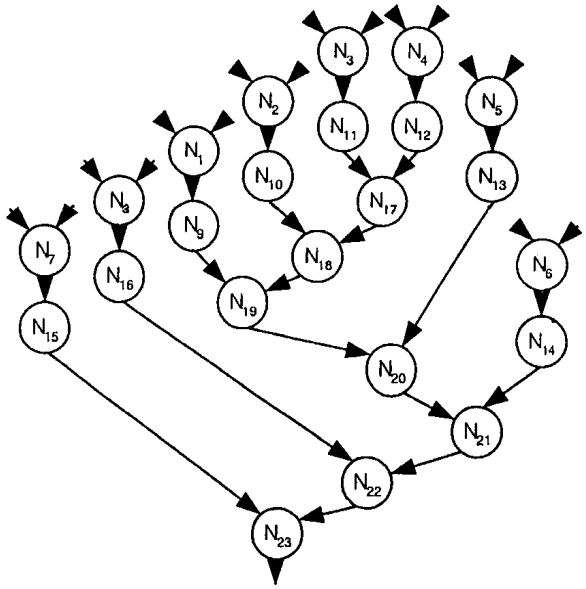
Fig. 9　ステップ4

5.　評価

提案手法の探索時間と解の精度を調べるために典型的なDFGのテンプレートであるFIRとEWを用い，PE数が4個の場合について評価を行う．

5.1　探索時間

提案手法は，2種類のDFGにおいて10^{-3}秒以下の探索時間となった．総当たり法ではノード数が10個のDFGで10^3秒程度の探索時間が必要であった．提案手法は総当たり法と比較した場合に高速であると言える．

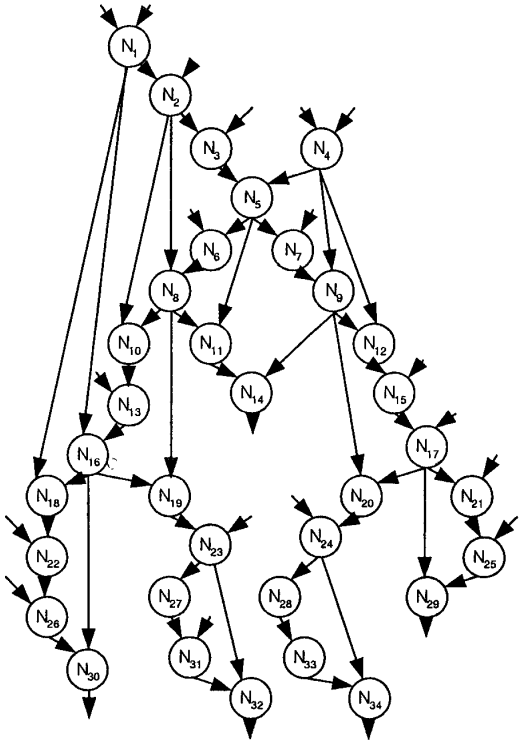


(a) FIR

	PE1	PE2	PE3	PE4
C.S.1	2	4		
C.S.2	3	12	1	
C.S.3	11	10	5	
C.S.4	17	9	6	
C.S.5	18	13	8	
C.S.6	19	14	7	
C.S.7	20	16		
C.S.8	21	15		
C.S.9	22			
C.S.10	23			

(b) Feasible Solution

Fig. 10　評価 1



(a) EW

	PE1	PE2	PE3	PE4
C.S.1	1			
C.S.2	2	4		
C.S.3	3			
C.S.4	5			
C.S.5	6	**7**		
C.S.6	8	9		
C.S.7	10	12		
C.S.8	13	15		
C.S.9	16	17		
C.S.10	19	20		11
C.S.11	23	24	**18**	14
C.S.12	27	28	22	21
C.S.13	31	33	26	25
C.S.14	32	34	30	29

(b) Solution

Fig. 11　評価 2

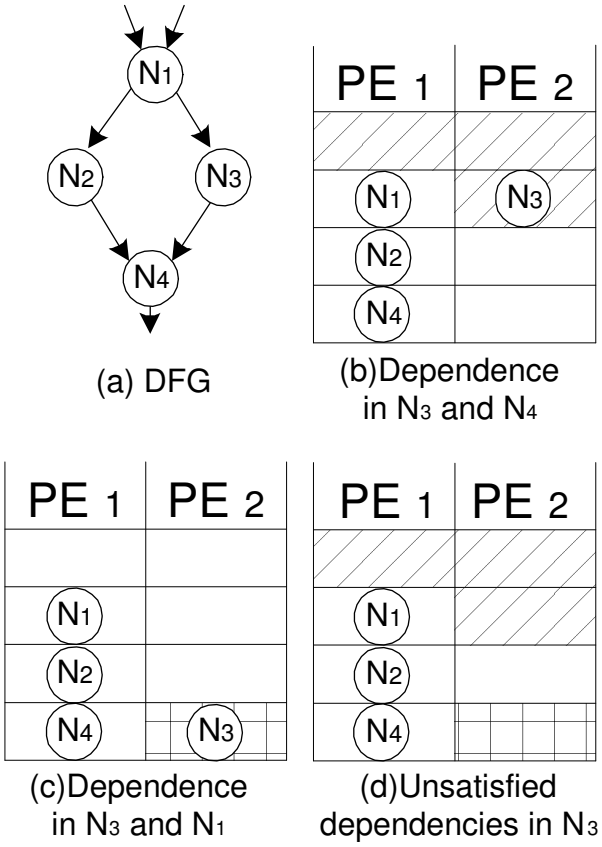


Fig. 12　可能解を得られない例

5.2　解の精度

FIRのDFGと結果をFig. 10に示す．FIRの場合，最適解を探索できた．表中の数字はノード番号を表している．

EWのDFGと結果をFig. 11に示す．EWでは可能解を探索できていない．Fig. 11の表中，下線付きのノードがDFGの依存関係を満たしていないためである．

可能解を探索できない原因について考察する．ここで，Fig. 12(a)のDFGについて，モジュール数を2個とし，アロケーションとスケジューリングを考える．モビリティは全てのノードで0になる．そこで，N_1・N_2・N_4をクリティカルパスとする．クリティカルパスをセルに配置した結果がFig. 12(b)である．最後に，残ったN_3を配置する．N_4とN_3との依存関係を満たすセルはFig. 12(b)網掛けのセルである．

一方，N_1とN_3との依存関係を満たすセルはFig. 12(c)の斜線のセルである．ここで，(a)よりN_3はN_1とN_4との依存関係を同時に満たす必要がある．このためには，(b)の斜線のセルと(c)の網掛けのセルの交わりのセルが必要である．しかしながら，(d)に示すように交わりのセルは存在しない．よってN_3を配置できない．従って，提案手法では，クリティカルパスの一部に複数のパスを有する場合，可能解を探索できていない．

6.　まとめ

条件分岐を有するDFGを対象とするリアルタイムハイレベルシンセシスについて，局所転送時間を最小にすることで処理時間全体を最小にする方法を提案した．しかしながら，クリティカルパスの一部に複数のパスを有する場合，可能解を探索できていない．このような場合を除き，提案手法は，最適解を高速探索できるなどの有効性を実験的に明らかにした．この方法は，条件分岐のないDFGに対しても有効である．

知能集積システム用VLSIの実現には，リアルタイムハイレベルシンセシスは重要であることから，提案手法の改良としてクリティカルパスの一部に含まれる2個以上のパスの一方に冗長ノードを入れる，あるいは分岐している部分のパス上にあるノードを特定の1個のPEにアロケーションするなどし，適用限界を拡大することが今後の課題である．

本研究は，科学研究費補助金の交付を受けて行った研究の成果である．

参考文献

1) 今井正治:ASIC技術の基礎と応用 ”亀山充隆，” 第8章：ロボットエレクトロニクス，118/140，電気情報通信学会(1994)

2) 亀山充隆, 藤岡与周：ロボット用VLSIプロセッサシステム, 日本ロボット学会誌, **14**-1, 22/25 (1996-1)

3) 工藤隆男, 亀山充隆：ロジックインメモリ構造に基づく知能集積システム用VLSIプロセッサのハイレベルシンセシス, 電気学会論文誌 C, **123**-8, 1374/1381 (2003)

4) Gajski, D., Dutt, N., Wu, A. and Lin, S.: HIGH-LEVEL SYNTHESIS Introduction to Chip and Sustem Design, 359, Kluwer Academic Publichers (1992).

5) 桜井貴康：総論－システムLSIのアプリケーションとシステムLSIの課題, 信学会誌, **81**-11, 1082/1086 (1998).

6) Kauts, W.H.:Cellular Logic-in-Memory Array, IEEE Trans. Computers, **18**-8, 719/727 (1969).

7) Hanyu, T. and Kameyama, M.:Multiple-Valued Logic-in-Memory VLSI Architecture Based on Floating-Gate-MOS-Pass-Transistor Logic, IEICE Trans. Electron, **28**-9, 1662/1668 (1999).

8) 工藤隆男, 羽生貴弘, 亀山充隆：ロジックインメモリアーキテクチャに基づく道路抽出VLSIプロセッサの構成, 計測自動制御学会論文集, **36**-11, 1009/1018 (2000).

9) 張山昌論, 工藤隆男, 亀山充隆：最適アロケーションに基づく道路抽出VLSIプロセッサとその高安全知能自動車への応用, 計測自動制御学会論文集, **J84-D-1**-6, 531/539 (2001).