C_006

2次割当問題に対するタブー探索法に基づくハードウェア解法

A Hardware Algorithm for Solving the Quadratic Assignment Problem Based on Tabu Search

木村 義洋* 若林 真一†
Yoshihiro Kimura Shin'ichi Wakabayashi

1 はじめに

2次割当問題 (Quadratic Assignment Problem, QAP) は NP 困難な組合せ最適化問題の1つであり，数理計画法等の通常の最適化手法では最適解を求めることは困難であることが知られている．このため，遺伝的アルゴリズムなどの各種の発見的手法が提案されている．一方，この問題は巡回セールスマン問題，VLSI レイアウト設計におけるセル配置問題など，工学分野の様々な問題に応用できるため，これまでに多くの研究が行われている [1].

QAP に対する有望な発見的手法の1つにタブー探索法に基づく解法がある [4]．タブー探索法は NP 困難な組合せ問題に対する頑健な発見的手法として知られている [2, 5]．QAP のベンチマークライブラリとして知られている QAPLIB[6] において，QAPLIB の多くのベンチマーク問題の最良解はタブー探索法に基づく手法で得られている．しかしながら，QAP の問題サイズが大きくなると，タブー探索法では探索解の総数が厖大になり，実用的な計算時間で優良解を得ることは困難になるという問題点がある．

本研究では，QAP に対するタブー探索法の計算時間に関する問題点を解決することを目的として，タブー探索法に基づく QAP に対する解法をハードウェアで実現し，FPGA 上に実現することを提案する．FPGA の大規模内部メモリを効率よく利用して複数の近傍解を並列処理により同時に評価し，かつ，各解に対する目的関数の評価をパイプライン処理で実行することで近傍解の評価時間が短縮可能になるため，従来のソフトウェア解法と比較して非常に短い実行時間でソフトウェア解法よりすぐれた解を生成可能となる．また，FPGA のプログラム可能性を利用することで，問題サイズと FPGA チップの規模を考慮した最適なハードウェア構成が利用可能になる．

以下では，まず，**2.** において QAP の定義を述べ，次に **3.** において QAP に対するハードウェア解法を提案する．さらに，**4.** において FPGA 上に実現した2次割当問題に対するタブー探索法に基づくハードウェア解法と同じ動作をするソフトウェアプログラムについて比較実験を行い，計算時間に関して評価した結果を示す．最期に **5.** で本論文をまとめる．

* 広島市立大学大学院情報科学研究科
† 広島市立大学情報科学部

2 準備

2.1 2次割当問題 (QAP)

2次割当問題 (QAP) とは，n 個の要素と n 個の場所が与えられたとき，目的関数を最小にする要素の場所への割当を求める問題である．すなわち，n 個の要素 $\{1,2,\ldots,n\}$ と2つの $n \times n$ 行列 $A=(a_{ij})$ と $B=(b_{ij})$ が与えられた時，式 (1) で表わされる目的関数 (割当コスト) を最小とする要素の場所への割当 π と，その時の割当コストを求める問題である．

$$F(\pi)=\sum_{i=1}^{n}\sum_{j=1}^{n}a_{ij}b_{\pi(i)\pi(j)} \quad (1)$$

ここで π は n 個の要素の順列を表す．行列 A は距離行列と呼ばれ，a_{ij} は場所 i と j の間の距離を示す．また，行列 B はフロー行列と呼ばれ，b_{kl} は要素 k から l までの物質の流れ（フロー）を表す．要素数 n を問題のサイズとよぶ．

2.2 タブー探索法

タブー探索法は，組合せ最適化問題を解くための発見的解法の一つであり，探索法の実行中は常に問題の許容解を保持し，許容解 x の近傍解の集合 $N(x)$ の中で，x 以外の最良の解を次の解として選び，この操作の繰り返しにより最適解を求めていく手法である [2]．単純に近傍解の中から最良解を選んでいくことにするとすぐに局所解に陥るので，タブー探索法では，最近探索した解をタブーリストと呼ばれるメモリに一定期間保存しておき，その解への再探索を一定期間禁止することで局所解からの脱出を可能にするという特徴がある．このため，選択された解が現在の解より目的関数を悪くするものであっても次の解への移動が強制される．タブー探索法の終了条件にはいくつかのものが知られており，例えば解の更新をあらかじめ決められた回数行った際に終了する，最良の評価値が一定期間更新されなくなったら終了する，等がある．

2.3 QAP に対するタブー探索法

タブー探索法を QAP に適用する場合，許容解は n 個の要素の順列 ϕ である．順列 ϕ に含まれる任意の異なる2要素 i と j を交換して得られた結果の順列を π とするとき，π は ϕ の近傍解であると定義する．定義より，サイズ n の許容解 ϕ の近傍解の集合は $n(n-1)/2$ の要素を持つ．これより，タブー探索法における現在の解から

近傍解への移動は，現在の順列 ϕ のユニットの対 (r,s) の交換で実現される．

許容解 ϕ のコストがすでに計算されている場合，ϕ の近傍解 π のコストを式 (1) で最初から計算する必要はなく，π のコストは交換したユニット r と s についての差分を求めることで計算可能である．r と s を交換する前の解を ϕ，交換した後の解（近傍解）を π とすると，コストの差分は式 (2) で与えられる．

$$\Delta(\phi,r,s)=\sum_{i=1}^{n}\sum_{j=1}^{n}(a_{ij}b_{\phi(i)\phi(j)}-a_{ij}b_{\pi(i)\pi(j)}) \quad (2)$$

ただし，$\pi(r)=\phi(s)$，$\pi(s)=\phi(r)$，$\pi(k)=\phi(k), 1\leq k\leq n, k\neq r, k\neq s$ である．もし，与えられる行列が対称行列で対角成分が全て 0 であるなら，式 (2) は以下のように書き換えられる [1]．

$$\Delta(\phi,r,s)=2\sum_{k\neq r,s}(a_{sk}-a_{rk})\times (b_{\phi(s)\phi(k)}-b_{\phi(r)\phi(k)}) \quad (3)$$

QAP の応用問題の多くは QAP に定式化すると与えられる行列は対称行列で対角成分が全て 0 となるため，本研究では差分式は式 (3) で表されると仮定する．

QAP に対するタブー探索法におけるタブーリストの構成としては，近傍解として採用した解をリストとして記憶することが考えられる．しかし，この方法ではリストに必要なメモリ容量が厖大になると共に，タブーリストの効率のよい探索が困難になる，という問題点がある．Skorin-Kapov はこの問題点を解消するため，以前の近傍解から現在の近傍解を構成するために交換したユニットの対をタブーとして記憶することを提案し，その有効性を実験的に示した [3]．また，Taillard も同じアイデアを用いて，効率のよいタブー探索を実現している [4]．そこで，本研究においてもタブーリストは近傍解を生成するために交換したユニット対で構成することとした．

3 提案ハードウェア

3.1 回路構成

本研究では，式 (3) の差分計算に並列処理やパイプライン処理を導入することにより，ソフトウェア解法より高速にタブー探索法を実行することを目指す．問題サイズ (n) が 16 の場合の提案ハードウェアのブロックダイアグラムを図 1 に示す．QAP に対する提案ハードウェアは主に差分計算ユニット (DCU) とタブーメモリユニット (TMU) から構成される．以下にそれぞれを説明する．

3.2 差分計算ユニット

回路は n 個の差分計算ユニット (DCU) を持ち，それぞれの DCU は式 (3) における各 $k(1\leq k\leq n)$ に対応した計算を行う．すなわち，行列 A と B の値が格納されているメモリから計算に用いる値を読み出し，減算器，乗算器に値を渡し，差分値を求める．行列 A と B は列ごとに分散して各ユニットに記憶されている．すなわち，k 番目の DCU には A の列データ A_{*k} と B の列データ

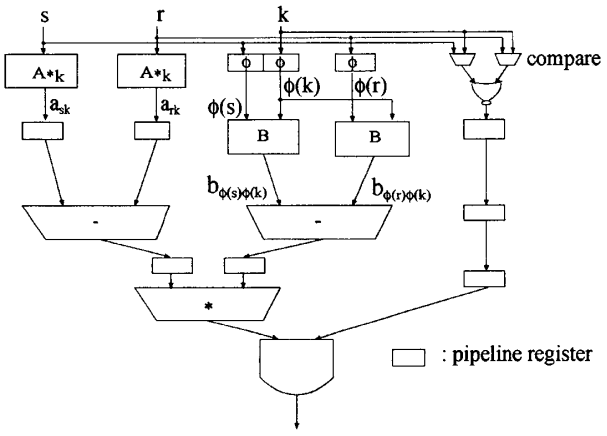

図 2: 差分計算ユニット

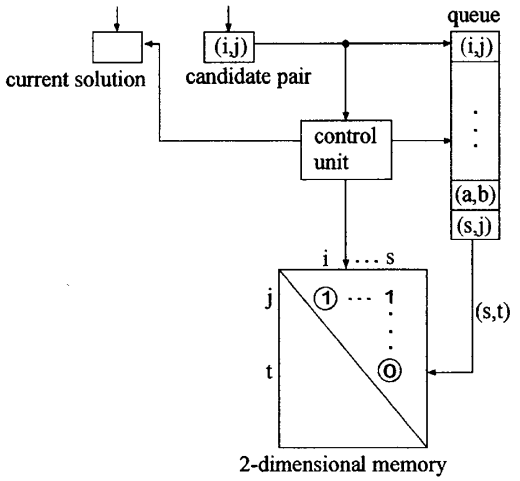

図 3: タブーメモリユニット

$B_{*\phi(k)}$ が格納されている．DCU のブロック図を図 2 に示す．

DCU の入力は交換するペア r，s である．また，各 DCU は固有の k の値を持つ．文字が記入されている四角がその文字が格納されているメモリであり，何も書かれていない四角はパイプラインレジスタである．台形は各演算器を示す．1，2 サイクル目で行列 A_{*k} が格納されているメモリから a_{sk}，a_{rk} を，ϕ が格納されているメモリから $\phi(s)$，$\phi(r)$ と $\phi(k)$ を読み出す．それと同時に k の値が r または s と同じかどうかを比較し，同じ場合は 1 を出力する．次の 2 サイクルでは，前の 2 サイクルで求めた $\phi(s)$，$\phi(r)$ と $\phi(k)$ を用いて $b_{\phi(s)\phi(k)}$ と $b_{\phi(r)\phi(k)}$ を求める．その次のサイクルで $a_{sk}-a_{rk}$ と $b_{\phi(s)\phi(k)}-b_{\phi(r)\phi(k)}$ を計算し，さらに次のサイクルで乗算を行う．最後に，k の値が r または s と同じ場合は差分計算を行わなくてよいので，計算結果を 0 にする．

r と s の値はカウンタにより $(1,2)$ から $(n-1,n)$ まで順次増加し，各 (r,s) についてパイプライン処理で式 (3) を計算する．

n 個の差分計算ユニットの出力は，その総和を求めた後で比較回路で順次比較され，差分が最も小さい (r,s)

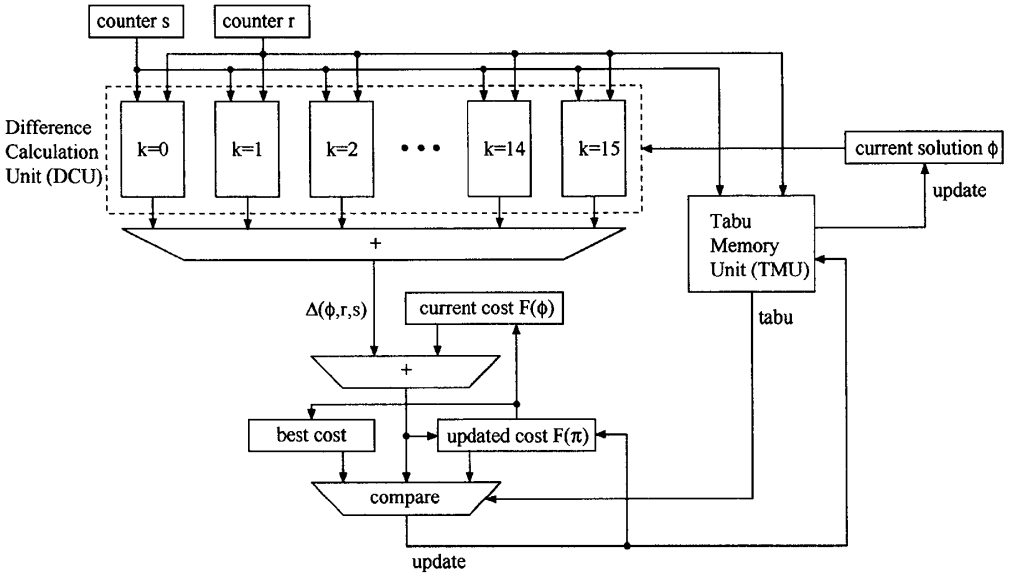

図 1: 提案ハードウェアのブロックダイアグラム ($n = 16$).

の対が次の許容解の候補として選択される (図 1 参照). 候補となった許容解はタブーメモリユニットでタブーリストに含まれているかどうかが判定され，含まれていなければ許容解として保持される．この比較回路もパイプライン動作をする．詳細は紙面の都合上，省略する．

近傍解が更新されると，それによって DCU に格納されている B の列データの内，交換されたユニット r と s に対応する列データを持つ DCU がそれぞれの B の列データを入れ換える．

3.3 タブーメモリユニット

タブーメモリは許容解の更新に伴って交換されたペアを順に一定期間格納するキュー (シフトレジスタで実現) と，キューに存在するペアのフラグを格納する二次元メモリで構成されている．ブロック図を図 3 に示す．

次の許容解の候補が見つかった場合，タブーメモリユニットには対応する添字の対 (i, j) が送られる．許容解がタブーかどうかの決定は，(i, j) をアドレスとする二次元メモリのデータ値 (フラグ) を調べる．データ値が 0 であればタブーでないので交換するペアをキューに入れ，キューに入れられたペアに対応する二次元メモリのデータ値を 1 とし，キューから出て行くペアに対応する二次元メモリのデータ値を 0 とする．

タブーリストをハードウェアで実現する場合，シフトレジスタだけでも実現は可能である．しかし，シフトレジスタだけで実現した場合，添字の対 (i, j) がリストに含まれているかどうかを定数時間で調べるためにはシフトレジスタを構成するすべてのレジスタに比較器を付ける必要があり，ハードウェアの規模が大きくなる．一方，最近の大規模 FPGA はかなり大規模なメモリを内部に持っている．このため，FPGA の内部メモリを有効に利用できる上記の方式を採用した．

3.4 FPGA による実現

提案ハードウェアをハードウェア記述言語 Verilog-HDL を用いて回路記述を行い，FPGA 設計ツールで論理合成，配置配線を行い，FPGA 評価ボードの上の FPGA にハードウェアとして実現した．FPGA 設計ツールには Altera 社の Quartus II Version5.0，FPGA 評価ボードには Power Medusa MU200-SX60 (ボード上の FPGA は Altera 社の EP1S60F1020C7) を用いた．利用した FPGA チップは 57,120 個の論理エレメント (LE) と，内部メモリとして 292 個の 4k ビットブロック RAM を登載している．FPGA チップのクロック周波数は 40MHz に設定した．FPGA 設計ツールは CPU が Pentium 4 2.4GHz，メインメモリ 2GB の PC 上で実行した．

4 実験的評価

QAP に対する提案ハードウェア解法を評価するため，ソフトウェア解法との比較実験を行った．比較対象のソフトウェア解法として，ハードウェア解法と同一のタブー探索法に基づく解法を C 言語でプログラムとして作成し，Sun Microsystems 社 Sun Blade 1500(CPU:UltraSPARC, 1062MHz, メインメモリ 2GB) のワークステーション上で実行した．

提案ハードウェア解法の評価実験においては，QAP の問題のサイズ n を 16, 32, 64, 128 とし，それぞれに対して許容解の更新回数を 100,000 回とし，ソフトウェア解法で解を求めた場合との計算時間の比較を行った．タブーリストのサイズ (長さ) については，予備実験を行った上で，最適なサイズと考えられる問題のサイズ n とした．QAP のインスタンス (問題) は QAPLIB[6] のベンチマークから選んだ．問題ごとに提案ハードウェア解法のハードウェア記述を用意し，回路として FPGA 上に実現した．

提案ハードウェア解法の論理合成結果を表 1 に，ハードウェア解法とソフトウェア解法の実行時間に関する実験結果を表 2 に示す．表 1 において，“LE 数” は回路の実現に要した FPGA の論理エレメント数である．“必

表 1: 論理合成結果

問題サイズ	Verilog-HDL 記述行数	合成時間 (s)	LE 数	必要メモリ (byte)
16	1800	164	2,109(1 %)	65,728(1 %)
32	2300	398	5,801(10 %)	524,724(10 %)

表 2: 実行時間に関する実験結果

問題サイズ	最良解	ソフトウェア (s)	ハードウェア (s)	速度向上比
16	68	26	0.3175	81.89
32	130	264	1.255	210.36
64	116	2153	5.02(予測)	428.88
128	64	17528	20.08(予測)	872.91

要メモリ" は回路で使用したメモリ量である．表 2 において，"最良解" は得られた最良解のコスト (目的関数の値)，"ハードウェア"，"ソフトウェア" はそれぞれハードウェア解法とソフトウェア解法の実行時間，速度向上比は 速度向上比 = (ソフトウェアの実行時間/ハードウェアの実行時間) である．なお，今回のハードウェア実現では設計の簡単化のため，各ユニットは行列 A と B のデータをすべて持つようにした．このため，FPGA チップの内部ブロック RAM の容量制約により，問題サイズが 64 と 128 の場合は実装ができなかった．このため，表に記載している問題サイズが 64，128 の場合のハードウェア解法の実行時間については，許容解の更新回数を 10,000 回として FPGA 設計ツール上で提案ハードウェア解法の RTL シミュレーションを行い，その結果からの予測値である．いずれの問題においても，QAPLIB で公開されている最良解が得られている．

実験結果より，ワークステーションの CPU のクロック周波数は FPGA のクロック周波数の約 26 倍であるのにも関わらず，ハードウェア解法はソフトウェアソフトウェア解法より圧倒的に高速であることがわかる．これは提案ハードウェア解法においては並列処理とパイプライン処理が有効に導入されているためだと考えられる．また，問題サイズの増加と共に速度向上比も増加している．これは，問題サイズの大きいほど，提案ハードウェアの並列度が上がるためだと考えられる．これらの結果から，提案ハードウェア解法の有効性は示された．

なお，問題サイズが 64 以上の場合も，3.2 節に記述したように行列 A と B の 1 列のデータのみを各 DCU に記憶させるようにすれば，問題サイズ 128 の場合も問題なく実現できると予想している．また，FPGA のクロック周波数については，回路のクリティカルパスの見直しなどをすることでより向上させることが可能になると予想している．

5 おわりに

本研究では，2 次割当問題に対するタブー探索法に基づくハードウェア解法を提案し，FPGA を用いて実現した．そしてソフトウェア解法と比較することにより提案ハードウェア解法の有効性を示した．最新の FPGA は今回実験で用いた FPGA の 3 倍以上の論理エレメント数を持っており，今後も FPGA の大規模化は進むと予測されているので，2 次割当問題の高速解法としての提案手法の有効性は高いと考えられる．

今後の課題としては，サイズの大きな問題に対する提案ハードウェア解法の実装，それに伴って増加するメモリサイズの削減等があげられる．また，与えられた問題サイズに応じて最適な回路構成のハードウェア記述言語による回路記述を自動生成することも興味深い課題である．さらに，QAP 以外の問題に対するタブー探索法のハードウェアによる実現も重要だと考えられる．また，タブー探索法においては長期記憶の有用性も知られているが [2]，ハードウェア実現が容易な長期記憶方式の開発も望まれる．

謝辞 本研究の実施にあたり，熱心に御討論いただきました広島市立大学情報科学部の永山忍博士に感謝いたします．本研究の一部は平成 18 年度科学研究費補助金基盤研究 (C)(課題番号 18500042) により行った．

参考文献

[1] E.Çela, *The Quadratic Assignment Problem: Theory and Algorithms*, Kluwer Academic Publishers (1998).

[2] F.Glover, M.Laguna, *Tabu Search*, Kluwer Academic Publishers (1997).

[3] J.Skorin-Kapov, "Tabu search applied to the quadratic assignment problem," *ORSA J. Comput.*, 2, 1, pp.33–45 (1990).

[4] E.Taillard, "Robust taboo search for the quadratic assignment problem," *Parallel Computing*, 17, pp.443–455 (1991).

[5] 柳浦睦憲, 茨木俊秀, 組合せ最適化: メタ戦略を中心として, 浅倉書店 (2001).

[6] http://www.seas.upenn.edu/qaplib/