社団法人 情報処理学会 研究報告
IPSJ SIG Technical Reports

2008－SLDM－133（12）
2008 / 1 / 16

並列リコンフィギャラブルプロセッサDAPDNA-2を用いた集合被覆問題の高速解法

石川 浩行† 清水 翔† 荒川 豊† 山中 直明† 斯波 康祐††

† 慶應義塾大学理工学部情報工学科 〒223-8522 神奈川県横浜市港北区日吉3-14-1
†† アイピーフレックス株式会社 〒141-0021 東京都品川区上大崎2-27-1 サンフェリスタ目黒6階
E-mail: †{ishikawa,shimizu,arakawa}@yamanaka.ics.keio.ac.jp, ††yamanaka@ics.keio.ac.jp, †††shiba@ipflex.com

あらまし 高速広帯域ネットワークの発展を背景に普及した，大容量の映像を配信するためのサービスにおいては，ユーザがオリジナルサーバからのみコンテンツをダウンロードした場合，オリジナルサーバの負荷が大きくなる．一方，オリジナルサーバ以外にもコンテンツを分散させる手法が提案されているが，コンテンツレプリカの最適配置計算は集合被覆問題に帰着され，NP困難である．そのためネットワーク規模が膨大になると，ソフトウェアで処理することは難しい．本研究では，レプリカ配置の全組合せを複数グループに最適分割してパイプライン処理を行う集合被覆問題の高速解法を提案する．IPFlex社が開発した並列リコンフィギャラブルプロセッサDAPDNA-2の評価ボードに実装し，特性評価の結果，動作クロック数が17倍のIntel Pentium 4と比較して実行時間が40倍以上改善することから，提案方式の有効性を示す．

キーワード リコンフィギャラブルプロセッサ，DAPDNA-2，集合被覆問題，レプリカ配置問題，NP困難

Fast calculation method of Set Cover Problem on parallel reconfigurable processor DAPDNA-2

Hiroyuki ISHIKAWA†, Sho SHIMIZU†, Yutaka ARAKAWA†, Naoaki YAMANAKA†, and Kosuke SHIBA††

† Dept. of Information and Computer Science, Faculty of Science and Technology, Keio University Hiyoshi 3-14-1, Kohoku-ku, Yokohama, 223-8522 Japan
†† IPFlex Inc. Kamiosaki 2-27-1, Sun felista Meguro 6F, Shinagawa-ku, Tokyo, 141-0021 Japan
E-mail: †{ishikawa,shimizu,arakawa}@yamanaka.ics.keio.ac.jp, ††yamanaka@ics.keio.ac.jp, †††shiba@ipflex.com

Abstract This paper proposes a fast calculation method of the set cover problem, which is implemented on reconfigurable processor DAPDNA-2 of IPFlex Inc. Content replication is a widely employed technique to improve the performance of large-scale content distribution systems. Replica placement method is derived from the set cover problem which is known to be NP-hard. It is difficult to caluculate the large-scale set cover problem on a program counter-based processor. Our proposed algorithm divides the combination optimally and performs pipeline operation. Experimental results show that the proposed algorithm reduces the execution time by 40 times compared to Intel Pentium 4 2.8GHz.

Key words Reconfigurable processor, DAPDNA-2, Set cover problem, Replica placement problem, NP-hard

1. はじめに

近年，FTTH (Fiber To The Home) などの高速広帯域ネットワークにおいてさまざまなサービスが利用されている．一例として，大容量の映像を配信するためのCDN (Contents Delivery Network) が挙げられる．CDNにおいては，ユーザがオリジナルサーバからのみコンテンツをダウンロードすると，オリジナルサーバの負荷が大きくなり，非効率である．そこで，オリ

ジナルサーバ以外のノードにコンテンツのコピーを配置することにより，アクセス速度を向上させる手法が提案されている．P2P オーバレイネットワーク上の各ノードは，自身がコンテンツを保有していない場合，検索リクエストを他のノードに転送する．トラヒックの増大を防ぐため，転送範囲には制限があり，全ノードの要求を満たすような効率的な配置を決定する問題をレプリカ配置問題と呼ぶ．

また，インターネットの動画配信では，複数の宛先を指定して，WDM (Wavelength Division Multiplexing) ネットワークの利用が今後普及する可能性がある [1]．WDM とは波長の異なる複数の光信号を同時に利用する技術であり，マルチキャストにおける効率的な波長割り当ては重要である．最小限の波長で全てのユーザにマルチキャストすれば，以降のコネクション要求が棄却されにくい．また，ネットワークの波長使用状況は頻繁に変わるため，高速に計算する必要がある．

レプリカ配置問題や，マルチキャストにおける波長割り当て問題は，数学的には集合被覆問題に帰着する．集合被覆問題とは，与えられた集合全体をカバーする部分集合で最小のものを探す問題であり，NP 困難である [2]．そのため，ノード数や波長数が増加すれば計算量も膨大になり，既存のプロセッサでは最適解を得ることが難しく，一般にはグリーディ法が用いられている [3]．グリーディ法は近似アルゴリズムの一種であり，集合被覆問題では最適解を常に得ることは不可能であると証明されている．

関連研究として，NP 完全である充足可能性問題の全解探索を FPGA で実装した論文 [4] が発表されているが，真理値割当に制限を加えており，しかも特性評価において Pentium 4 などの汎用プロセッサと比較を行っていないため，その有効性は明確ではない．

そこで本研究では，IPFlex 社 [5] が開発した並列リコンフィギャラブルプロセッサ DAPDNA-2 を用いて並列計算及びパイプライン処理を行うことにより，全ての組合せを列挙する集合被覆問題の高速解法を提案する．DAPDNA-2 は，32bit の専用 RISC プロセッサ (DAP) と DNA と呼ばれる並列処理エンジンの，大きく 2 つから構成される．DNA には PE(Processing Element) と呼ばれる演算器が，マトリックス状に合計 376 個配置されている．複数の PE を接続することでデータフローマシンを生成し，並列演算が可能である．

提案方式では，全ての組合せを複数グループに分割して，DAPDNA-2 でパイプライン計算を行う．組合せを 2 進数で表現すると，大小関係を設定することができ，Beeler らが考案したアルゴリズム (以下，Beeler 法と呼ぶ) [6] を実行することで，昇順に全ての組合せを生成する．組合せを複数グループに分割するにあたり，解決すべき問題が 2 つある．1 つは，各グループの先頭パターンを求める方法である．Beeler 法で生成可能な組合せは，一見不規則に変化しており，先頭から任意の順番のパターンを求めることは困難である．もう 1 つは，全体の計算クロック数が最小になるような分割数を求める方法である．この最適な分割数は，組合せの総数や，Beeler 法の計算クロック数に依存する．本研究では，各グループの先頭パターンを求めるアルゴリズムを提案し，全体の計算クロック数が最小になるような最適な分割数を理論的に求めることにより，これら 2 つの問題を解決する

さらに評価ボード DAPDNA-EB4 上に実装し，Pentium 4 2.8GHz と計算時間を比較した．逐次的に全ての組合せを生成する手法では，計算時間のオーダーは $O({}_nC_k)$ となるのに対し，提案方式では $O(\sqrt{{}_nC_k})$ に抑制でき，組合せの総数がある一定値を超えると計算時間が逆転することを示す．

2. 関連研究

2.1 レプリカ配置問題

インターネットにおける大規模で効率的なコンテンツ配布を実現する技術としてレプリカ配置の研究がなされている．各ノードにはストレージコスト，更新コスト，QoS 要求が定められており，全ノードの QoS 要求を満たしつつ，合計コストが最小になるようなレプリカ配置を求めなければならない [7]．ストレージコストとは，ノードにレプリカを配置するのにかかるコストであり，ハードディスク容量が大きいほど小さい．更新コストとはオリジナルサーバからの距離である．オリジナルサーバのコンテンツが更新された場合，各レプリカサーバに更新データを送信する必要があるため，レプリカサーバはオリジナルサーバに近い方が望ましい．ここで，数学的に問題を定式化する．ネットワークを $G=(V,E)$ と表現し，オリジナルコンテンツを持っているノードを $r \in V$，レプリカを配置したノードの集合を R とする．ノード v のストレージコスト，更新コスト，QoS 要求をそれぞれ $s(v), u(v), q(v)$ とし，ノード u, v 間の距離を $d(u,v)$ とする．ノード v にレプリカが配置されていない場合，v は最寄のレプリカサーバにコンテンツ検索リクエストを転送するが，その距離は $q(v)$ 以内でなければならない．全ノードの QoS 要求を満たすためには，以下の式を満足する必要がある．

$$\min_{w \in R \cup r} d(v,w) \leqq q(v) \tag{1}$$

この条件を満たした上で，$\sum_{v \in R} s(v)+u(v)$ が最小になるレプリカ配置を考える必要がある．レプリカ配置問題は，集合全体をカバーする部分集合で最小のものを探す，集合被覆問題に帰着される．集合被覆問題の定義は以下の通りである．

最小重み集合被覆問題

全体集合 U と U の部分集合の族 $\mathbf{S}$ に対して，$\bigcup_{R \in \mathbf{R}} R = U$ を満たす部分族 $\mathbf{R} \subseteqq \mathbf{S}$ で重みが最小のものを求める．

最小重み集合被覆問題の最適解は NP 困難であることが証明されているため，レプリカ配置問題も NP 困難である [8]．そのため様々なグリーディ法が考案されているが，常に最適解を得ることは不可能であると数学的に証明されており，最適解を得るためには組合せの全てのパターンを調べる以外方法はない．しかし，レプリカ配置問題におけるノード数が増加した場合，計算量が膨大になり，既存のプロセッサでは全ての組合せを実用的な時間で調べることは困難である．

2.2 組合せ生成アルゴリズム

n 個の数 $1, 2, \ldots, n$ から k 個選ぶ ${}_nC_k$ 通りの組合せを全て生

成するアルゴリズムが M.Beeler, R.W.Gosper, R.Schroppel によって考案された．このような組合せは 2 進法 n 桁の数で表すことができる．右端のビットから順に $1,2,\ldots$ に対応させれば，例えば $n=8$ の場合，$(2,3,4,6)$ は 00101110 で表せる．このような表し方をすれば，組合せに順序関係を定めることができる．例えば 00101110 < 00110110 であるから $(2,3,4,6)<(2,3,5,6)$ という順序になる順序関係で最初の組み合わせは 00001111 である．このアルゴリズムは，00001111 から 11110000 まで全ての組合せを順番に生成することが可能である．詳細なアルゴリズムを以下に示す．

（1） ある組合せ X の 1 を最も右のものだけ残して，他を全て 0 にしたものを $Smallest$ とする．

（2） $Ripple = X + Smallest$

（3） $Ripple$ の 1 を最も右のものだけ残して，他を全て 0 にしたものを $NewSmallest$ とする．

（4） $NewSmallest/Smallest$ を 1bit 右シフトして 1 を引いたものを $Ones$ とする．

（5） $Ripple$ と $Ones$ のビット毎の論理和が，X の次の組合せである．

$n=6, k=3, X=001110$ の場合のアルゴリズムの実行例を以下に示す．

（1） $Smallest = 000010$

（2） $Ripple = X + Smallest = 010000$

（3） $NewSmallest = 010000$

（4） $NewSmallest/Smallest = 001000$ であり，1bit 右シフトすると 000100 なので，$Ones = 000011$

（5） $Ripple$ と $Ones$ のビット毎の論理和は 010011 となり，これが X の次の組合せである．

[定理 1] Beeler 法は ${}_nC_k$ 通りの組合せを昇順に全て生成する．

[証明 1] ある組合せ X の次の組合せ Y を求めるとき，X の 1 のビットで最も右のものに着目する．ここから $m(\geq 1)$ 個の 1 が連続して並んでいるとしても一般性を失わない．これら m 個の 1 の集合を S_1 とし，それ以外の $k-m$ 個の 1 の集合を S_2 とする．また，S_1 のうち最も左の 1 は最下位ビットから $a(\geq 1)$ ビット目にあるとする．

このとき，次の組合せ Y は，X の S_1 のうち最も左の 1 が，最下位ビットから $a+1$ ビット目で，それより右の $m-1$ 個の 1 を最下位から順に並べたものであり，S_2 は変わらない．

$\because$) 背理法によって証明する．Y の S_1 のうち最も左の 1(以下，l) が最下位ビットから $a+1$ ビット目にないと仮定する．l が $a+1$ ビット目より左と仮定すると，この組合せが Y より小さくなるためには，1 の合計数 k は不変のため，S_2 に属する 1 のどれかを右に移動しなければならない．しかし，この組合せは X より小さくなり矛盾する．

次に l が $a+1$ ビット目より右と仮定すると，この組合せが X より大きくなるためには，S_2 に属する 1 のどれかを左に移動しなければならない．しかし，この組合せは Y より大きくなり矛盾する．

したがって，S_1 のうち最も左の 1 が最下位ビットから $a+1$ ビット目にあり，このような組合せのうち最小のものが Y である．

$Ripple$ は Y の最下位から連続した $m-1$ 個の 1 を 0 にしたものである．$NewSmallest/Smallest = 2^m$ であり，1 ビット右シフトすると 2^{m-1}

$\therefore\ Ones = 2^{m-1} - 1 = \underbrace{11\cdots1}_{m-1}$

$Ripple$ と $Ones$ でビット毎に論理和を計算すると，Y が得られる．□

3. 提案方式

3.1 概　　要

組合せの全てのパターンを高速に求めることは，レプリカ配置問題だけでなく，波長割り当て問題など，集合被覆問題に帰着可能な問題に適用できる．提案方式では，全ての組合せを複数グループに分割して，DAPDNA-2 でパイプライン計算を行う．各グループの先頭パターンを 1 クロックずつずらして順次 Beeler 法の計算回路に入力する．例えば，組合せを均等に 4 分割した場合，最後に入力されたグループの結果は，最初に入力されたグループの結果よりも 3 クロック遅延するが，全体の計算時間はおよそ 1/4 になる．ここで，2 つの問題が生じる．1 つは，組合せを複数グループに分割したとき，各グループの先頭パターンをどうやって求めるか，という問題である．Beeler 法は，組合せを昇順に求められるが，一見不規則に変化しており，任意の順番のパターンを求めることは困難である．もう 1 つは，全ての組合せをいくつに分割すれば，全体の計算クロックが最小になるかという問題である．この最適な分割数は，組合せの総数や，Beeler 法の計算クロック数に依存する．提案方式では，これら 2 つの問題を解決する．

3.2 最適な組合せ分割数

先頭から任意の順番のパターンを求める処理を a クロック，Beeler 法により次のパターンを求める処理を b クロックとする．組合せを分割せずに，n 個から k 個を選ぶ組合せを全て求めるには $b({}_nC_k-1)$ クロックである．組合せを 2 分割して提案方式を適用すれば $a+\frac{b({}_nC_k-1)}{2}+1$ クロックかかる．組合せを 3 分割して提案方式を適用すれば $2a+\frac{b({}_nC_k-1)}{3}+2$ クロックかかる．同様に x 分割すると，計算クロック数 y は

$$\begin{aligned} y &= (x-1)a + \frac{b({}_nC_k-1)}{x} + x - 1 \\ &= \frac{b({}_nC_k-1)}{x} + (a+1)x - a - 1 \end{aligned}$$

クロックである．y が最小になるときの x の値を求めればよい．相加相乗平均の関係より，

$$\begin{aligned} y &= \frac{b({}_nC_k-1)}{x} + (a+1)x - a - 1 \\ &\geqq 2\sqrt{\frac{b({}_nC_k-1)}{x}(a+1)x} - a - 1 \\ &= 2\sqrt{b({}_nC_k-1)(a+1)} - a - 1 \end{aligned}$$

等号成立は $\frac{b({}_nC_k-1)}{x} = (a+1)x$ すなわち

$$x = \sqrt{\frac{b({}_nC_k - 1)}{a+1}} \qquad (2)$$

のときである．

3.3 任意の順番のパターンを求める計算法

本研究では，Beeler 法で昇順に生成される組合せ群の先頭から任意の順番のパターンを求める方法を提案する．一般に次式が成り立つ．

$${}_nC_k = \sum_{i=k-1}^{n-1} {}_iC_{k-1} \qquad (3)$$

ここで i, j を非負整数として，${}_iC_j$ は，異なる i 個のものから異なる j 個のものを選ぶ組合せの総数である．先頭から m 番目の組合せを求めたい場合，まず

$$\sum_{i=k-1}^{x_1} {}_iC_{k-1} \geqq m \quad (x_1 \leqq n-1)$$

を満たす最小の x_1 を見つける．1 は全部で k 個のため，${}_{x_1}C_{k-1}$ とは，最上位の 1 が x_1 ビット目で，0 から $x_1 - 1$ ビットのうち $k-1$ 個が 1 である組合せを意味する．すなわち，求める組合せの x_1 ビット目は 1 である．

そして先頭から m 番目は，${}_{x_1}C_{k-1}$ の中では $m - \sum_{i=k-1}^{x_1-1} {}_iC_{k-1}$ 番目に相当するため，

$$m \rightarrow m - \sum_{i=k-1}^{x_1-1} {}_iC_{k-1}$$

と置換する．次に

$$\sum_{i=k-2}^{x_2} {}_iC_{k-2} \geqq m \quad (x_2 \leqq x_1 - 1)$$

を満たす最小の x_2 を見つける．${}_{x_2}C_{k-2}$ とは，最上位の 1 が x_2 ビット目で，0 から $x_2 - 1$ ビットのうち $k-2$ 個が 1 である組合せを意味する．すなわち，求める組合せの x_2 ビット目は 1 である．同様の操作を k 回繰り返すと $x_1, x_2, \cdots, x_k$ が得られ，対応するビットを 1 にしたものが，先頭から m 番目の組合せになる．

例えば，${}_6C_3$ の先頭から 6 番目のパターンは以下のように求められる．

$${}_6C_3 = {}_2C_2 + {}_3C_2 + {}_4C_2 + {}_5C_2 = 1 + 3 + 6 + 10$$

先頭から 6 番目は，${}_4C_2$ に含まれるため，次にこれに (2) 式を適用する．

$${}_4C_2 = {}_1C_1 + {}_2C_1 + {}_3C_1 = 1 + 2 + 3$$

先頭から 6 番目は ${}_4C_2$ の中では 2 番目に相当するため，${}_2C_1$ に含まれることがわかる．

$${}_2C_1 = {}_0C_0 + {}_1C_0 = 1 + 1$$

${}_2C_1$ の中では 1 番目に相当するため，${}_0C_0$ である．そして，右から数えて 4，2，0bit 目を 1 にしたパターンが先頭から 6 番目である．

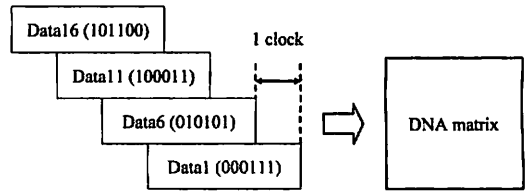

図 1 パイプライン演算

3.4 DAPDNA-2 への実装

本研究では，DAPDNA-2 上で動作するアプリケーションプログラムを開発するための統合開発環境 DAPDNA-FWII を使用して，提案アルゴリズムを実装した．ノードの選択方法に関して，例えば全ノード数が 6 で，そのうち 3 ノードにレプリカを配置する場合，000111 から 111000 まで昇順に組合せを全て求める必要がある．DAPDNA-2 では 32 ビットデータを扱うため，32 ノードを上限として設計した．さらに各ノードもビットデータで表現し，ノード i の QoS 要求を満たせば i ビット目を 1 にする．(1) 式において，ノード w のデータの w ビット目と v ビット目は 1 であり，このとき w は v をカバーすると言う．レプリカ配置問題では，3 つの 32 ビットデータについてビット論理和を計算し，その値が 111111 になれば全てのノードをカバーしたことを意味する．例えば，101001 というノード選択を行う場合，ノード 0，ノード 3，ノード 5 が候補となる．QoS 要求に関して，以下の式が成り立つと仮定する．

$$d(4,0) \leqq q(4), \quad d(1,3) \leqq q(1), \quad d(2,5) \leqq q(2)$$

このとき，ノード 0 は 010001，ノード 3 は 001010，ノード 5 は 100100 と表現される．ビット論理和を計算すると 111111 となるため，この組合せは全ノードをカバーすることができる．全ノードをカバーする組合せが複数存在する場合は，その中からコストが最小の組合せを選択すればよく，それが最適解である．

提案方式ではまず最適な分割数を計算するが，それ以降は主に以下の 3 つの処理に分割される．

（1） 先頭から m 番目のパターンを求める．

（2） Beeler 法により，次のパターンを求める．

（3） 対応するデータを取得して，全ノードがカバー可能か判定する．

実装では，処理 (1) を DAP で計算し，その結果をメインメモリに格納し，そのアドレスを DNA に受け渡す．DNA は指定アドレスからデータを読み取り，処理 (2) および処理 (3) をパイプラインで実行し，最終的な計算結果をメインメモリに格納する．図 1 に ${}_6C_3$ を 4 分割した場合のパイプライン演算の流れを示す．20 通りの組合せを 4 分割するため，先頭から 1，6，11，16 番目の組合せを求め，DNA matrix でパイプライン演算を実行する．

1 チップで 4 並列演算を行っており，これを 2 チップ使用することにより 8 並列演算を実現している．8 並列演算を行った場合の実行クロック数は以下の通りである．

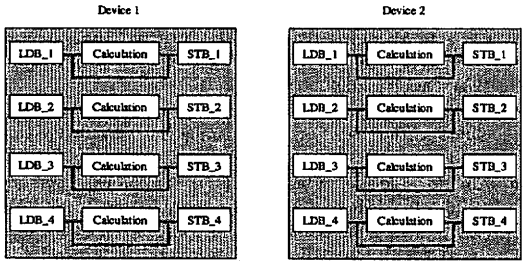

図 2　マルチプロセッサにより 8 並列演算が実行可能

$$y = 2\sqrt{b\left(\frac{{}_nC_k}{8} - 1\right)(a+1)} - a - 1$$

図 2 にマルチプロセッサにおける実装例を示す．本研究では 2 チップ使用して提案方式を実装したが，チップ間の計算を完全に独立に行うことにより，高速な演算を可能とした．1 チップにはそれぞれ 4 個のロードバッファ(LDB) とストアバッファ (STB) があり，メインメモリからのデータの読み書きを行う．

(3) 式の最適な分割数を d とした場合，配列に d 個のデータを格納し，各 LDB から 8 等分したデータを読み出す．すなわち，Device 0 の LDB_1 において a[0] から a[d/8-1] をロードし，LDB_2 において a[d/8] から a[d/4-1] をロードする．また今回の実装では，Beeler 法を実行するのに 33 クロック必要とするが，1 つの LDB から読み出すデータ数を 33 以上にするためには，演算回路に遅延を挿入して計算クロック数を増加させる必要がある．各 LDB においてロードするデータ数を減らすことで，全体の実行時間に占めるロード時間の割合を小さくするだけでなく，挿入する遅延を少なくすることが可能となる．

4. 特性評価

特性評価では，IPFlex DAPDNA-2 166MHz における提案方式と，動作クロックが 17 倍の Intel Pentium 4 2.8GHz における従来の組合せ全探索を計算時間に関して比較する．DAPDNA-2 の評価には評価ボード DAPDNA-EB4 を用いた．

図 3 にノード数に対する実行時間を示す．ここでは，全ノードから 8 ノードを選択する組合せを全て求めている．実測値と理論値に多少誤差が認められるが，実行時間の増加傾向はほぼ等しいことがわかる．提案方式では，ノード数が増加しても実行時間があまり増加せず，30 ノードのとき，実行時間が 40 倍以上改善している．これは，DAPDNA-2 の並列化及びパイプライン処理によって同時並列的に計算しているためである．

図 4 に全ノードの 1 割のノードにレプリカを配置した場合の理論上の実行時間を示す．先頭から m 番目の組合せを求める処理を a クロック，Beeler 法により次のパターンを求める処理を b クロックとする．また，実装での測定値より $a = 330, b = 33$ とした．全ノード数 n に対して，k ノードにレプリカを配置した場合，Pentium 4 および DAPDNA-2 における理論上の実行時間をそれぞれ t_c, t_p とすると以下のように表される．

$$t_c = \frac{b({}_nC_k - 1)}{2.8 \times 10^9}(sec)$$

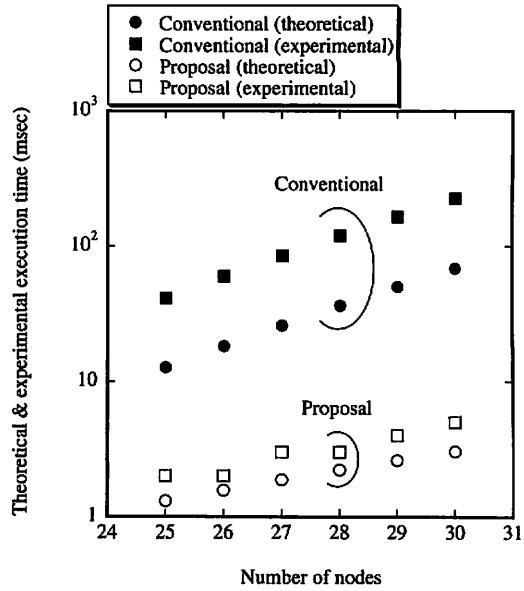

図 3　ノード数に対する実行時間

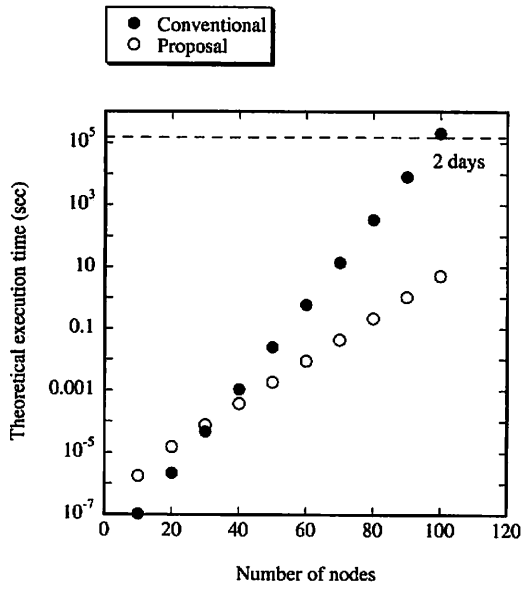

図 4　組合せの総数に対する実行時間比較 (理論値)

$$t_p = \frac{2\sqrt{b({}_nC_k - 1)(a+1)} - a - 1}{166 \times 10^6}(sec)$$

100 ノードの場合，従来方式では 2 日間もの計算を要するが，提案方式の計算量は $O(\sqrt{{}_nC_k})$ であるため，実用的な時間内で計算可能である．したがって提案方式の方がネットワークの規模に関してスケーラビリティーがあると言える．

図 5 に分割数に対する DAPDNA-2 の実行時間を示す．提案方式は最適な分割数を求めることができ，25 ノードにおいて 328 分割，27 ノードにおいて 472 分割の場合，実行時間が最小となる．パイプライン処理では，計算結果が 1 クロックずつず

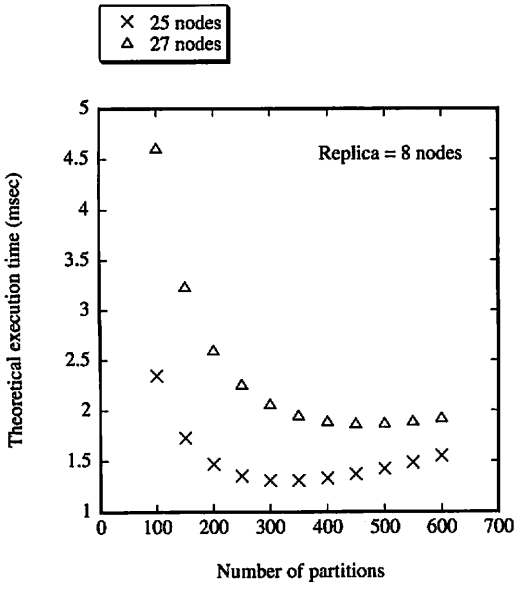

図 5 分割数に対する実行時間比較 (理論値)

れて出力されるため，分割数を大きくすると実行時間は減少するが，ある値を超えると増加に転じる．

5. 結　　論

本研究では，組合せを複数グループに最適分割してパイプライン処理を行う，集合被覆問題の高速解法を提案した．CDN におけるレプリカ配置問題は集合被覆問題に帰着され，NP 困難であるため計算量が膨大になると，ソフトウェアで処理することは難しく，一般にはグリーディ法が用いられている．しかしながら，集合被覆問題ではグリーディ法を用いて常に最適解を得ることは不可能であると証明されているため，実用的とは言いがたい．そこで全ての組合せを高速に求めるために，組合せを複数グループに分割したとき，各グループの先頭パターンを求めるアルゴリズムを提案し，全体の計算クロック数が最小になるような最適な分割数を理論的に求めた．

IPFlex 社が開発した並列リコンフィギャラブルプロセッサ DAPDNA-2 の評価ボードに提案方式を実装し，Pentium 4 と実行時間を比較した．提案方式の計算量は $O(\sqrt{n C_k})$ であるため，実用的な時間で計算可能であり，ネットワークの規模に関してスケーラビリティーがあると言える．特性評価の結果，Pentium 4 と比較して実行時間が 40 倍以上改善することから，提案方式の有効性を示した．

謝辞 本研究は日本学術振興会科学研究者補助金「並列処理による高速経路探索に基づいた次世代光・IP 連携ネットワークの研究」によって行われた．また，グローバル COE プログラム「アクセス空間支援基盤技術の高度国際連携」(C12) の助成を受けたものである．関係者各位に深謝する．

文　　献

[1] Jianping Wang, Biao Chen, R.N.Uma, "Dynamic Wavelength Assignment for Multicast in All-Optical WDM Networks to Maximize the Network Capacity," IEEE Journal on selected areas in communications, vol.21, No.8, pp.1274-1284, October 2003.

[2] M.R.Garey, D.S.Johnson, Computers and Intractability: A Guide to the Theory of NP-Completeness. W.H. Freeman and Company, 1979.

[3] Alberto Aloisio, Vincenzo Izzo, Salvatore Rampone, "FPGA Implementation of a Greedy Algorithm for Set Covering," Real Time Conference, pp.446-450, June 2005.

[4] 山岸洋平, "FPGA を用いた NP 完全問題の全解探索法に関する研究," 北陸先端科学技術大学院大学 情報科学研究科情報処理学専攻 修士論文, 2003.

[5] アイピーフレックス株式会社 (http://www.ipflex.com)

[6] M.Beeler, R.W.Gosper, R.Schroeppel, HAKMEM (http://www.inwap.com/pdp10/hbaker/hakmem/hakmem.html)

[7] Hsiangkai Wang, Pangfeng Liu, Jan-Jan Wu, "A QoS-Aware Heuristic Algorithm for Replica Placement," Grid Computing 7th IEEE/ACM International Conference, pp.96-103, September 2006.

[8] Xueyan Tang, Jianliang Xu, "QoS-Aware Replica Placement for Content Distribution," IEEE Transactions on parallel and distributed systems, vol.16, No.10, pp.921-932, October 2005.