ハイパフォーマンス
コンピューティング 92－5
（2002．10． 25）

並列直接解法によるSmoothed Aggregation MG法の改良と評価

藤井昭宏† 西田 晃† 小柳義夫†

本稿では，Smoothed Aggregationに基づくAlgebraic Multigrid Method（AMG法）の並列アルゴリズム[13]の計算量を分析し，各PEに固定の行列サイズを割り当てた場合にレベル数一定，反復回数が一定で収束するならば，問題サイズによらず各PEは一定の計算量，通信量ですむことを示す．但し，異方性のある問題や複雑形状の問題では，もっとも粗いレベルで直接解法をする必要があり，その場合は各PEの計算量は最も粗いレベルの問題サイズに依存してしまう．そこで最も粗いレベルの問題サイズが大きくても対応できるように並列直接解法を適用した場合の計算量を考察する．数値実験ではクラスタ上で最大1500万次元の3次元ポアソン方程式を解き，問題サイズによらずほぼ一定の時間で収束することを確認する．また10000倍の3次元異方性問題を解き，反復回数や実行時間を分析する．

Improvement and evaluation of Smoothed Aggregation MG with a parallel direct solver

Akihiro Fujii,† Akira Nishida† and Yoshio Oyanagi†

In this paper, we analyze the complexity of parallel AMG[13] and show that each PE has constant amount of computation and communication irrelevant of the problem size on some assumptions. We have to use direct method on the coarsest level for problems which has complex geometry or anisotropy. In that case, each PE's amount of computation depends on problem size. We evaluate the effect of a parallel direct solver for the coarsest level. We have tested with three-dimensional Poisson problems which have up to 15 million nodes ($250\times250\times250$) and anisotropic problems which have 10000 times anisotropy on a workstation cluster.

1. はじめに

近年，楕円型2階の偏微分方程式を離散化した大規模連立一次方程式 $Ax=b$ の解法としてマルチレベルな解法が多く研究されている．そのような手法の一つにAMG法がある．AMG法の特徴として以下の4つがあげられる．

- 逐次計算の場合，収束までの計算量は $O(n)$ である
- 並列性がある
- 不規則行列に対しても有効である
- 行列データしか参照しない

AMG法の中の最も有力な解法の一つにSmoothed Aggregation Multigrid[2)3)]法がある．問題行列が与えられるとその行列から次元数の少ない別の行列を作る．その行程を繰り返し複数の行列をつくり出し，それらを利用して効率良く解く手法である．

Smoothed Aggregation MG法は，問題行列の構造を点枝接続行列と見なしたときのグラフを使用する．グラフの節点は問題行列の各行に対応し，枝は問題行列の非ゼロ要素に対応する．そのグラフ内である節点の近くの節点をまとめたものをアグリゲートといい，次の粗いレベルの節点に対応するものである．このアグリゲートというものを利用して，次の粗いレベルの問題行列を生成する．

並列アルゴリズムとしては[13]に従い実装した．高並列な環境を考慮し，PE間でアグリゲートを共有をしない手法である．アルゴリズムとして並列性が高く通信が少ないが，粗いレベルでの行列の取扱が問題になる．特に高並列な環境の場合，PE間でアグリゲートを共有をしないため，各PEに最低1節点以上次元が残る．すなわち最も粗いレベルでの次元数が増大し，そのレベルの解を求めるのが困難になる．特に異方性のある問題については，粗いレベルの行列であっても反復法で解を求めるのは難しく，直接解法を適用しなければならない．

そこで粗いレベルで疎行列並列直接解法を適用することが考えられるが、それについてPSPASESライブラリ[14]を利用して実装し、評価を行う．

† 東京大学大学院 情報理工学系研究科 コンピュータ科学専攻
Dept. of Computer Science, Graduate School of Information Science and Technology, University of Tokyo

はじめに異方性のない問題に対して，レベル数一定で，一定反復回数 (v-cycle) で収束するならば，PE 数と問題サイズを比例させれば実行時間が一定になることを示す．但し，レベル数一定ということはもっとも粗いレベルの次元数が問題サイズに応じて増大し，一定反復回数で収束しにくくなることを意味する．次に並列直接解法を適用する場合の問題サイズと計算量について考察する．

数値実験では大規模な 3 次元ポアソン方程式（最大 1500 万次元）や 10^4 倍の異方性のある問題に対して GeoFEM ライブラリ[8] に含まれるソルバと性能比較を行ない，有効性を検証する．

2. Smoothed Aggregation MG 法

マルチグリッド（MG）法は問題方程式を複数のレベルに離散化して解く手法である．各レベルでは次元数の異なる問題行列が生成される．複数レベルの問題行列を利用することにより，それぞれの次元数に対応した誤差周波数成分を取り除くことができる．その結果，誤差周波数成分を均等に減らすことができ，収束までの反復回数が問題サイズに非依存の解法になる．

Smoothed Aggregation Multigrid 法は問題行列の階層構造の生成部（問題生成部）と，それを用いての反復計算部（反復部）に分けられる．図 1 にアルゴリズムを書く．

但し $LEVEL$, $MAXCOUNT$ はそれぞれ最大レベル数と反復回数を表す定数．各レベルの緩和法 S_{lev}，問題行列 A_{lev}，右辺ベクトル b_{lev}，解 u_{lev}，レベル lev からレベル $lev-1$ への補間 (prolongation) 演算子 P_{lev} とする．lev が大きいほど，粗いレベル，すなわち対応する問題行列の次元数が小さくなるとする．

始めに，問題 $A_1u_1 = b_1$ が与えられる．問題行列生成部では，粗いレベルを次々生成し，行列の階層構造を作成する．反復解法部ではそれを用いて V サイクルを行う．

2.1 問題行列生成部

問題行列生成部では A_{lev} から次のレベルの行列 A_{clev} を生成する． レベル間 (レベル $clev$ とレベル lev) の補間行列 P_{clev} を生成し，行列積 $A_{clev} = P_{clev}A_{lev}P_{clev}^T$ 計算する．

2.1.1 補間行列 P の生成

図 1 内の番号順に書く．

(1) 問題行列の非ゼロ要素のうち，対角に比して閾値以上のものを取り出す

(2) アグリゲート生成

(3) P_{clev} 生成

(1) は，問題行列 A の非ゼロ要素のうち値が小さいものは無視し

$$\tilde{A} = filter(A)$$

を計算する．$filter()$ は，行列を引数として対角に対して絶対値の小さい非対角要素 $|a_{ij}| < \theta * |a_{ii}| * |a_{jj}|$ を 0 にする操作である．但し θ は 0 から 1 の間の定数とする．

```
/* 問題行列生成部 */
for lev = 1 to LEVEL − 1 do
  clev = lev + 1
  /* (1) */
  Ã_lev = filter(A_lev)
  /* (2) */
  P̃_clev = aggregation(Ã_lev)
  /* (3) */
  P_clev = smooth(P̃_lev)
  A_clev = P^T_clev A_lev P_clev
end for

/* 反復解法部 (v cycle) */
for itercnt = 1 to MAXCOUNT do
  for lev = 1 to LEVEL − 1 do
    clev = lev + 1
    S_lev(A_lev, b_lev, u_lev)
    b_clev = P^T_clev (b_lev − A_lev u_lev)
  end for
  S_LEVEL(A_LEVEL, b_LEVEL, u_LEVEL)
  for lev = LEVEL − 1 to 1 do
    clev = lev + 1
    u_lev = u_lev + P_clev u_clev
    S_lev(A_lev, b_lev, u_lev)
  end for
end for
```

図 1 Smoothed Aggregation MG アルゴリズム

(2) $\tilde{A}$ に基づくグラフ上で考える．節点は $\tilde{A}$ の各行，枝は非ゼロ要素に対応する．節点全体をアグリゲートと呼ばれる節点集合に分解する．アグリゲートは次の粗いレベルで 1 つの節点に対応し，グラフ上である節点を中心に近くの節点をまとめた節点集合と定義される．多くの場合ある節点から枝 1 本以内で到達できる節点集合となる．

以下の条件を満たすように全節点集合をアグリゲートに分けていく．

- 任意の節点がどこかのアグリゲートに属する
- アグリゲート同士は同じ節点を共有しない

例えば 5 点差分により生成された行列があったときはそれぞれのアグリゲートは 5 節点になる．ここで行列 $\tilde{P}_{clevel}$ を作成．同じくらいの大きさのアグリゲートを整然とつくることができるかどうかによって収束性が変わる．

$$\tilde{P}_{clevel}(i,j) = \begin{cases} 1 & \text{節点 i} \in \text{アグリゲート j} \\ 0 & \text{それ以外の場合} \end{cases}$$

(3) ではアグリゲートの節点に適切に重みづけを

する．$\tilde{P}_{clevel}$ をそのままレベル間の補間行列として使ってもよいが，収束性を高めるために緩和法を一回適用する．緩和法としては減速ヤコビ法を用いる場合が多い．減速ヤコビ法を用いた場合，ω を係数，D_{level} を行列 A_{level} の対角部として

$$P_{clevel} = (I - \omega D_{level}^{-1} A_{level})\tilde{P}_{clevel} \qquad (1)$$

と書ける．

2.1.2 行 列 積

補間行列 P を生成後，$A_{clev} = P_{clev}^T A_{lev} P_{clev}$ を処理し次レベル行列 A_{clev} を得る．

2.2 反復解法部

通常の MG 法と同様である．例としてレベル数が2のときを取り上げる．

2レベルの場合：細かいレベルでの緩和法で残った残差を粗いレベルに移す．粗いレベルでその残差を打ち消すような補正解を計算し，細かいレベルへ補正解を足しこむ．

以上のような手順を繰り返すことにより，それぞれのレベルに対応する誤差周波数成分を同時に効率良く減らすことができる．

反復解法部で行なうことは行列ベクトル演算とベクトルとベクトル加減算のみであり，並列性が高い．

3. 並列アルゴリズムと計算量

3.1 並列アルゴリズム

データの分割は節点に基づく領域分割をする．各 PE に分割された問題行列が渡され，それに基づいて粗いレベルの行列も生成していく．粗いレベルの領域分割は初めに行われた領域分割に従う．

並列にアグリゲートを生成する手法については，多くの研究[1),4)]がなされている．各 PE が担当節点集合を独立にアグリゲートを生成する手法や，最大独立点集合を並列に求めるアルゴリズム[12)]に基づく手法，領域間の境界部分をはじめに通信しあってアグリゲートを共有する手法などである．本研究では並列性や通信量を考慮しアグリゲート生成を各 PE 間で独立に行っている．

並列アルゴリズムで必要になる主な通信は，以下の2点となる．

- アグリゲートの生成，緩和法を処理後，PE 境界にあるアグリゲートを隣接 PE に通信 (行列 P) 以後 この通信処理を COMM1 と呼ぶ．
- 行列積 P^TAP を各 PE 担当領域ごとに独立に計算後，担当 PE へ通信．以後この通信処理を COMM2 と呼ぶ．

問題行列生成部は問題サイズによらず一定のレベル数を生成するとする．最も粗いレベルについて反復解法を適用する場合は，問題行列生成部は問題サイズに依存しないことを示す．その後で，並列直接解法を用いた場合にどの程度問題サイズに依存するか考察する．

3.2 問題行列生成部の計算量

[13)] における問題行列生成部が，各 PE に割り当てられる行列サイズと各行の平均非ゼロ要素数にのみ依存することを示す．これにより，それぞれの PE に割り当てられる行列サイズ，1行の非ゼロ要素数が固定の場合，PE 数や問題サイズによらず問題行列生成部は一定時間で終ることが言える．

問題行列生成部は 複数レベルを生成する時間であり，1レベル生成する時間が各 PE に割り当てられた行列サイズと各行の平均非ゼロ要素数にのみ依存することが言えれば良い．

1レベル作成するための並列実行時の計算量を考える．行列生成部の並列実行時の計算量は，関数 filter，aggregation，smooth，P^TAP，COMM1，COMM2 を算出すればよい．各 PE に割り当てられる行列の次元数を M，1行の平均非ゼロ要素数を E とする．

3.2.1 各処理の演算量

関数 filter は非ゼロ要素を処理するため，$O(E*M)$．

関数 aggregation はアグリゲートをなるべく隣り合うように生成していく．各節点にはスコアがあり，スコアの最も高いものを次のアグリゲートの中心となる節点とする．スコアはプライオリティキューにより管理する．すなわち，節点を選択したのち、それと連結している節点はプライオリティキューから削除し，距離2の節点のスコアを増やす．すなわち，プライオリティキューを操作する回数は節点数 (M 回) で抑えられる．アグリゲート数は $M/(E+1)$ と見なすことができる．計算量は節点の探索 $(M/(E+1)*E^2)$ とプライオリティキューの操作の合計で，$O(M*E + M\log M)$ となる．スコアの初期設定で PE 間の領域境界部分を高くすることで，領域境界部分から順にアグリゲートの生成をする．スコアを用いずランダムにアグリゲートを生成することにより $O(M*E)$ とできるが，特に高並列な環境の場合は収束が悪くなると考えられる．

関数 smooth は生成したアグリゲートに緩和法をかける．アグリゲートは，サイズ $E+1$ の節点集合と考えることができる．アグリゲート一つの緩和法に必要な演算量は $(E+1)^2$，アグリゲート数は $M/(E+1)$ とすると，必要な演算量はそれらの積である $(E+1)M = O(E*M)$ となる．

行列積 P^TAP は行列 A の非ゼロ要素を基準に考える．行列 A の非ゼロ要素数は $E*M$ となる．A(i,j) 要素は行列 P の j 行目と i 行目の掛け合わせのときに適用される．すなわち，行列 P の1行の平均非ゼロ要素数を D とすると積は $D^2*E*M < O(E^3*M)$ 回行われる．但し D は E より小さくなることが多い．

COMM1 は PE 間のオーバラップ領域 (節点数 $E*M$ 以下) での通信であり，COMM2 は行列積 P^TAP の一部を通信する．すなわち，COMM1，COMM2 ともに E と M の式で抑えられる．

3.2.2 全体の計算量

問題行列生成部の演算量はEとMで抑えられ，レベルを生成するたびに，Mは少なくとも1/2倍になり，Eは一定で抑えられると仮定すると問題サイズに依らないことがわかる．問題サイズによらず生成するレベル数は一定のためである．

各レベルでの緩和法が機能して，反復回数が一定で収束するならば，反復解法部も問題サイズに依らず，一定時間で処理できる．

但し 反復回数一定で収束するためには，各レベルでの緩和法が有効に機能していなくてはならない．特に最も粗いレベルにおいては問題サイズに応じて次元数が大きくなっても正確に解く必要がある．

複雑形状の問題や，異方性の問題では最も粗いレベルを直接法で解かないと収束が悪くなることが知られている．またアルゴリズム[13]では領域分割ごとに一定数のノードが残ることになり、PE数が多くある高並列な環境では最も粗いレベルでも次元数が大きいものが残ると考えられる．そこで次セクション以降では，適用する疎行列並列直接解法の概略と全体の計算量の変化を分析する．

4. 並列直接解法

本稿では疎行列直接解法のライブラリであるPSPASES[14]を利用した．以下に概略を示すが，詳細は[15][16]を参照する．対象の行列として対称正定値行列を仮定している．直接解法は4つの段階に分割され処理される．ordering, symbolic factorization, numeric factorization, triangular solveである．ordering, symbolic factorizationでは，並列実行が可能なようにオーダリングの変更後，フィルインの場所を特定し行列のelimination treeを作成する．このオーダリングの変更に領域分割のライブラリであるMETISライブラリが利用されている．次にそのelimination treeにしたがってnumeric factorizationを行う．この段階で実際にLU分解が行われる．次にtriangular solveで前進後退代入をして解を求めることになる．

3次元のグラフで各節点における次数が一定の行列について計算量は表1のようになる．

表1 N節点 次数一定のグラフに対する計算量

	serial	parallel
order	$O(N \log N)$	$O(\frac{N \log N}{p^{1/2}})$
symbolic fact.	$O(N^{4/3})$	$O(\frac{N^{4/3}}{p}) + O(\frac{N^{2/3}}{p^{1/2}})$
numerical fact.	$O(N^2)$	$O(\frac{N^2}{p}) + O(N^{4/3}p^{1/2})$
tri. solve	$O(N^4/3)$	$O(\frac{N^{4/3}}{p}) + O(N^{2/3})$

表1から最も処理時間がかかるのが,numerical factorizationの部分である．PEに割り当てる次元数を固定にして並列化したとき，問題サイズとPE数が比例することになる．しかし，それでも$O(N^{11/6})$となり最も粗いレベルの問題サイズに大きく依存することになる．

問題サイズとPE数が比例すると仮定すると，triangular solveは，$O(N^{2/3})$で解けることになる．問題サイズが増加すると，反復解法部でも1反復当たりの時間が増えることになる．

5. 数値実験と評価

並列AMGアルゴリズムをCG法の前処理として実装し評価する．比較のためGeoFEM[8]ライブラリのソルバと同じインタフェースで実装し，ICCGソルバ[7]と比較を行った．問題サイズに対する反復回数や収束までの時間の変化に注目する．AMG法の反復計算部では 各レベルで対称ガウスザイデル法1回を各PE領域内で行ない，領域間ではヤコビ法を行なう．最も粗いレベルでは，isotropic problemsでは各レベルと同じ反復法を10回行ない，anisotropic problemsでは並列直接解法を適用した．テスト環境にはSun Blade 1000（UltraSPARCIII 750MHz ×2, 1GB主記憶）128ノードをMyrinet 2000により接続したクラスタを用いた．ノード間通信はMPICH-GM[6](version 1.2.1)により行なった．

5.1 isotropic problems

問題は3次元ポアソン方程式で以下のものである．

$$-(\frac{\partial}{\partial x}(\frac{\partial P}{\partial x}) + \frac{\partial}{\partial y}(\frac{\partial P}{\partial y}) + \frac{\partial}{\partial z}(\frac{\partial P}{\partial z})) = 0$$

問題領域はXYZ軸に垂直な面を持つ直方体とする．また境界条件については

$$\begin{cases} Xmin: & \frac{\partial P}{\partial x} = 10.0, \\ Ymin: & \frac{\partial P}{\partial y} = 5.0, \\ Zmax: & \frac{\partial P}{\partial z} = 1.0, \\ Zmin: & P = 0, \end{cases}$$

とし，その他の境界は自然境界条件とした．$Xmin$はX座標が最小の面を表す．1PE当たりの節点数を$50 \times 50 \times 50$に固定して残差の2ノルムが初期残差の10^{-13}倍になるまでの時間を計測する．問題行列生成部では，どの場合も4レベル生成するとする．表2に問題サイズとPE数，収束までの時間を書く．

表2と図2から問題サイズに依らずほぼ一定の時間で解けていることが分かる．PE数が少ない場合は行列の分散が少ないために，通信量や行列積の計算量が少ないためより早く終了している．27PEより大きいときはほぼ一定時間で終了している．これは以下のような理由による．

- レベル数が一定
- 収束までの反復回数が一定

このような条件が成立する範囲においてはほぼ一定時間で収束すると考えられる．

各レベルでの担当節点数は125000, 7386, 405, 9のように変化した．

表 2 isotropic problems:問題サイズと AMGCG,ICCG 計算時間 (秒)

PE 数	問題サイズ	反復回数	AMGCG	反復解法部 (AMGCG)	ICCG
2PE	50×50×100	21	100.3	34.29	66.8
4PE	50×100×100	23	105.6	38.15	85.7
8PE	100×100×100	26	126.7	43.58	99.2
12PE	100×100×150	28	137.1	48.15	124
18PE	100×150×150	29	154.9	51.61	145
27PE	150×150×150	29	158.7	52.20	159
32PE	100×200×200	30	159.0	53.84	176
48PE	150×200×200	30	165.0	54.83	203
64PE	200×200×200	30	168.4	56.46	212
125PE	250×250×250	30	184.3	56.46	212

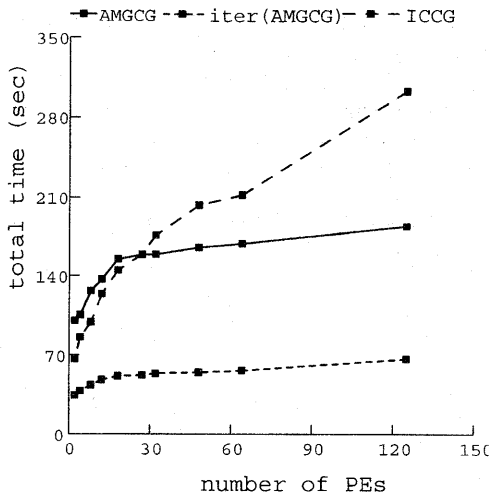

図 2 AMGCG と ICCG の実行時間：iter (AMGCG) は AMGCG の 反復解法部にかかった時間

5.2 anisotropic problems

前セクションと同じ 3 次元ポアソン方程式に Z 方向に 10^{-4} 倍の異方性を付けた問題を対象とする．異方性のある問題では，最も粗いレベルにおいて並列直接解法を適用する．この問題は反復解法では解くのが難しい問題である．

等方性の問題と比較すると異方性の問題の場合，粗くするときに次元数が減りにくい特性があるので各 PE に割り当てる節点数を少なめに設定する．今回の実験では，各 PE には $32 \times 32 \times 32$ の問題サイズ割り当てた．全体のレベル数は 3 とした．各 PE が担当する行列サイズは 32000, 3700, 550 というように変化した．すなわち最も粗いレベルにおける問題サイズは 550×PE 数になる．

表 3 により，この問題に非常に有効に働いていることが分かる．但し問題サイズが増えることにより，行列生成部の処理時間が $O(n^{11/6})$ で増加するため，このままの手法で高並列化は問題がある．

6. おわりに

本稿では Smoothed aggregation に基づく AMGCG 法の並列アルゴリズム[13] について計算量と通信量を評価した．一定のレベル数を生成するのであれば、問題行列生成部における各 PE の計算量と通信量は各 PE に割り当てられた行列サイズと各行の平均非ゼロ要素数にのみ依存し，問題サイズに依存しないことを示した．これにより，各 PE が担当する行列サイズが固定であれば，反復回数が一定で収束する場合は，問題サイズに非依存で一定の時間で収束する．125PE まで各 PE での問題サイズ固定で実行時間を計測しほぼ一定時間で終了することを確認した．

また異方性の問題については，

- 最も粗いレベルにて直接解法をしなければ反復回数が増大してしまうこと
- 各 PE 領域を独立に扱っているため最も粗いレベルでも次元数が問題サイズに応じて大きくなってしまうこと

という背景があり，最も粗いレベルにて疎行列並列直接解法[14] を利用し評価した．PE 数，問題サイズが増えると、最も粗いレベルの問題サイズも増加し問題行列生成部は一定時間では終了しないが，ある一定の問題サイズ以内であれば，異方性問題にも有効な解法であることを確認した．

今後の課題としては，

- 複雑形状の問題を解く
- ベクトルの問題にも対応する
- もっとサイズの大きい異方性の問題について実験する
- ベクトル化

など多くのことがあるが，ベクトルの問題への対応を中心に研究をすすめていく予定である．

参考文献

1) Ray S.Tuminaro, and Chales Tong: "Parallel Smoothed Aggregation Multigrid : Aggregation Strategies on Massively Parallel", SuperComputing 2000 Proceedings, 2000

2) P. Vanek, M. Braezina, and J. Mandel: "Con-

表 3 anisotropic problems:問題サイズと AMGCG,ICCG 計算時間 (秒)

PE 数	問題サイズ	反復回数	AMGCG	反復解法部 (AMGCG)	ICCG
2PE	32 × 32 × 64	32	30.8	11.2	125.8
4PE	32 × 64 × 64	28	32.5	10.5	170.9
8PE	64 × 64 × 64	30	36.8	12.3	180.5
16PE	64 × 64 × 128	32	41.6	14.0	343.0

vergence of Algebraic Multigrid Based on Smoothed Aggregation", Tech. Rep. report 126, UCD/CCM, Denver, CO, 1998.

3) P. Vanek: "Algebraic Multigrid Based on Smoothed Aggregation for Second and Fourth Order Problems", Computing, 56(1996), pp. 179-196

4) Van Emden Henson, and Ulrike Meier Yang, "BoomerAMG: a Parallel Algebraic Multigrid Solver and Preconditioner", Lawrence Livermore National Laboratory technical report, UCRL-JC-141495

5) G. Haase, M. Kuhn, and S. Reitzinger: "Parallel AMG on Distributed Memory Computers", SIAM Journal on Scientific Computing, accepted (2002).

6) http://www.myri.com/scs/

7) Kengo Nakajima, and Hiroshi Okuda: "Parallel Iterative Solvers with Localized ILU Preconditioning for Unstructured Grids on Workstation Clusters", IJCFD, 1999, Vol. 12, pp. 315-322.

8) http://www.geofem.tokyo.rist.or.jp

9) Tony F. Chan, and Petr Vanek: "Multilevel algebraic Elliptic Solvers." UCLA Math Dept CAM Report 99-9, February 1999. Invited paper, in Proc. of High Performance Computing and Networking 1999, Amsterdam, April 1999, Lecture Notes in Comp. Sci. 1593, Springer, pp. 1001-1014.

10) Andrew J. Cleary, Robert D. Falgout, Van Emden Henson, Jim E. Jones, Thomas A. Manteuffel, Stephen F. McCormick, Gerald N. Miranda, John W. Ruge: " Robustness and Scalability of Algebraic Multigrid" SIAM Journal on Scientific Computing 2000 Volume 21, Number 5 pp. 1886-1908.

11) Andy Cleary, Rob Falgout, Van Emden Henson, and Jim Jones: "Coarse-Grid Selection for Parallel Algebraic Multigrid", Proceedings of the Fifth International Symposium on Solving Irregularly Structured Problems in Parallel, Springer-Verlag Lecture Notes in Comp. Sci.

12) Mark Adams: "A Parallel Maximal Independent Set Algorithm", Proceedings 5th copper mountain conference on iterative methods,1998.

13) 藤井 昭宏, 西田 晃, 小柳 義夫: "領域分割による並列 AMG アルゴリズム", HPC91-6,Aug.2002.

14) http://www-users.cs.umn.edu/mjoshi/pspases/

15) Anshul Gupta, George Karypis, and Vipin Kumar: "Highly Scalable Parallel Algorithms for Sparse Matrix Factorization ",IEEE Transactions on Parallel and Distributed Systems Volume 8, Number 5.

16) Anshule Gupta, Fred Gustavson, Mahesh Joshi, George Karypis,and Vipin Kumar: "Design and Implementation of a Scalable Parallel Direct Solver for Sparse Symmetric Positive Definite Systems", Proceedings of the Eighth SIAM Conference on Parallel Processing, March 1997.