Vol.2013-HPC-138 No.9
2013/2/21

実時間シミュレーションへの応用を前提としたSMW公式を用いた逆行列計算のハイブリッド並列処理の評価

松井 祐太[1]　福間 慎治[1]　森 眞一郎[1]

概要：本論文では実時間時系列シミュレーションの高速化を目的とし，まず最初に連立方程式 $Ax = b$ の求解問題に対して，係数行列 A の近似行列の逆行列が既知の場合に，SMW 公式を用いて A の逆行列を求めて高速に求解する手法について解説し，ハイブリッド並列化について議論している．また，並列度による計算時間と通信時間の関係のモデル化を行っている．さらにハイブリッド並列の有効性の評価とモデルの検証を行っている．その結果，ハイブリッド並列処理を用いることで，メモリバンド幅の違う一部の計算機環境，次元数の違う行列データを用いた場合でもハイブリッド並列処理の有効性を示している．また，計算時間および通信時間と並列度の関係がモデルに従うことを示している．

1. はじめに

本研究では，多くのシミュレーション中に現れる線形方程式 $Ax = b$ の求解問題に着目する．特に，シミュレーション中のユーザからのインタラクションや時間経過によって係数行列 A 並びに右辺ベクトル b が微小変化を繰り返す状況でのリアルタイムシミュレーションの実現を研究のターゲットとする．図 1 は本研究が想定する時系列シミュレーションの典型的なシナリオ例である．上から下に向けてタイムステップが進行し，各行にはそれぞれのタイムステップにおいて解くべき線形方程式が示されている．この例では，各シミュレーションステップ毎に右辺ベクトルが変化するのに対して，左辺の係数行列は数ステップ毎に変化している．構造変形シミュレーションを例にとって考えると，非破壊変形が数ステップ起った後に破壊変形が起る現象が繰り返し発生している状態に対応している．なお，本論文では係数行列 A は対称行列を仮定し，時系列シミュレーション中には解くべき問題のサイズは不変かつ係数行列のランク落ちは発生しないものと想定する．

なお，実時間インタラクションを伴う時系列シミュレーションにおいては，１タイムステップのシミュレーションに要する時間とは無関係に現実世界の時間は進行する．そのため，ユーザが一定時間（例えば１分間）シミュレーションを行った場合に，何ステップのシミュレーションが行われたかは一意に定まらないという点は注意が必要である．

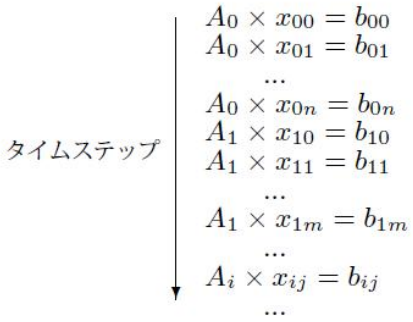


図 1　想定する時系列シミュレーションの典型的シナリオ

このようなシミュレーションを行う計算機資源としては，近年マルチコアプロセッサを搭載したコンピュータを複数台並列接続するハイブリッド型の並列処理環境が広く用いられており，計算に利用可能な計算機資源の並列度は今後ますます増大していく．問題サイズが非常に大規模なシミュレーションでは問題サイズを大きくすることによる Weak Scaling による並列性抽出が可能である．しかしながら，実時間性を要求するアプリケーションでは問題サイズを大きくしたいという要求はあるものの実時間性の確保が第一義であるため，Strong Scaling が有効な範囲での並列性抽出が主体となり，計算自体の並列化という従来の高速化手法のみでは，あり余る計算機資源を有効利用した更なる高速化が期待できない．

本論文では，このような実時間時系列シミュレーションの高速化を目的とし，まず最初に SMW 公式を用いた逆行

[1]　福井大学大学院　工学研究科

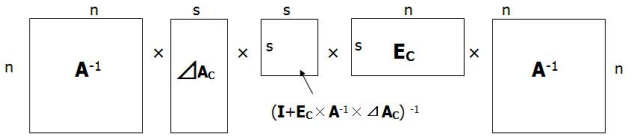


図 2 SMW 公式で用いられる行列の行列サイズ

列計算に基づく線形方程式求解問題のハイブリッド並列処理による高速化について報告し，次に，計算時間と並列度との関係のモデル化を試み，複数の環境での評価結果を報告する．

2. 研究の背景

2.1 SMW 公式

通常，連立一次方程式の求解問題では係数行列 A の逆行列を直接求めて解を導出することはまずない．しかしながら，A の近似行列の逆行列 $(A^{-1})'$ が既知であり，これを用いて高速に A^{-1} を求める方法が存在すれば A の逆行列を導出して方程式の解を求める方法も十分に有効な手法となる．

このような目的で使用可能な数学公式の 1 つに SMW 公式 (Sherman-Morrison-Woodbury Formula)?がある．

サイズ $n \times n$ の行列 A，サイズ $n \times s$ の行列 B，ならびにサイズ $s \times n$ の行列 C を考える．ただし，$n \geq s$ とする．A の逆行列を A^{-1} とすると，SMW 公式は次式で与えられる．

$$(A+BC)^{-1} = A^{-1} - A^{-1}B(I+CA^{-1}B)^{-1}CA^{-1} \tag{1}$$

いま，A が時間経過等によって微少変化した時の行列 A' を $A' = A + \Delta A$ と表現する．さらに ΔA を $\Delta A = \Delta A_c E_c$ とし，ΔA_c は ΔA から全ての要素が 0 である列を取り除いた行列，E_c は列の除去に対応して 0 と 1 からなる行列とする．この時，式 (1) において $B = \Delta A_c$，$C = E_c$ とすると，

$$\begin{aligned}(A')^{-1} &= (A+\Delta A)^{-1} \\ &= A^{-1} - A^{-1}\Delta A_c(I+E_cA^{-1}\Delta A_c)^{-1}E_cA^{-1}\end{aligned} \tag{2}$$

となる．式 (2) は $(A')^{-1} = A^{-1} - (\text{補正項})$ と見ることができ，変化後の行列 (A') の逆行列が A^{-1} の補正により導出できることを表している．式 (2) において，右辺第二項で行う行列積計算に使用する行列サイズの概要を図 2 に示す．なお，ΔA_c の列の数を s とする．ここで，行列 $(I + E_cA^{-1}\Delta A_c)^{-1}$ は図 2 の中央部分にある大きさ $s \times s$ の行列に対応し，この逆行列を求める計算量は $O(s^3)$ となる．その他の行列積の計算での計算量は $O(sn^2)$ となる．しかしながら，変化する要素数が微少 $(n >> s)$ であると仮定すると，計算量は $O(sn^2)$ である．

1. $S_1 = A^{-1} \times \Delta A_c$
2. $S_2 = E_c \times S_1$
3. $S_3 = I + S_2$
4. $S_4 = (S_3)^{-1}$ を直接法で計算
5. $S_5 = E_c \times A^{-1}$
6. $S_6 = S_4 \times S_5$
7. $S_7 = S_1 \times S_6$
8. $S_8 = A^{-1} - S_7$
9. $S_9 = S_8 \times b$

図 3 従来法の計算ステップ

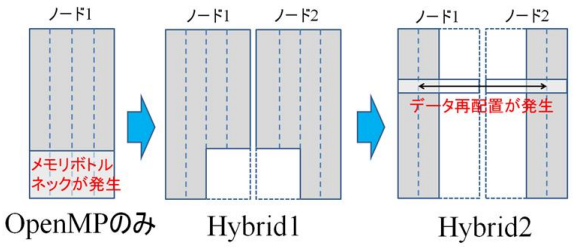


図 4 各手法の並列化方法の違い

2.2 OpenMP によるノード内並列化

SMW 公式での計算行程では行列積計算が多くの割合を占め，並列処理による高速化が得られる．また，係数行列 A の要素のうち時間変化する要素の数が微小であれば，$n >> s$ であるとともに A が密行列か疎行列かによらず，ΔA は疎行列となり，ΔA_c，E_c も疎行列となる．そこで，E_c および ΔA_c を圧縮形式で保存するとともに，零要素に対する演算を省略することで，高速化が得られる．我々の過去の研究?では，SMW 公式を 3 の 9 つのステップに分けて逆行列計算の実装を行い，マルチコアプロセッサを用いたノード内スレッド並列による並列処理および上述した疎行列性の利用により，良好な結果を得られた．これを従来法とする．

しかしながら，逆行列計算時間全体の 95%をステップ 7，8 が占めていることがわかった．また，ステップ 7，8 において並列度が上がっても十分な並列化効果が得られなかった．解析の結果，これは，並列化に伴い主記憶アクセスのバンド幅不足に起因していることが推測できた．

3. ハイブリッド並列処理

前述の問題を解決するために，ノード内スレッド並列だけでなく，複数の計算ノードを用いてノード間並列を行い，利用できるメモリバンド幅を増やすことによる高速化効果について検討する．手法として，以下の節ではメモリボトルネックを解消することを主眼とした Hybrid1 と，計算全体を並列処理する Hybrid2 を述べる．従来法と Hybrid1 および Hybrid2 の違いを図 4 に示す．

3.1 Hybrid1

Hybrid1?はメモリボトルネックとなっている部分のみ

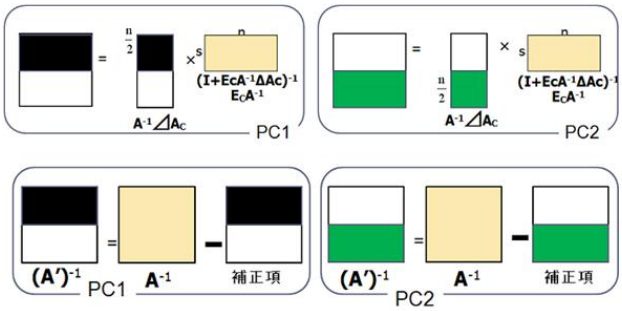


図 5 並列化部分の行列積モデル

を複数ノードで実行するものであり，その他の部分については重複実行を行うものとする．具体的には，S_7 において，$A^{-1}\Delta A_c$ の計算を行方向ブロック分割により並列処理する．その結果，S_8 においても行方向ブロック分割が行われた状態で計算が行われる．並列度 $p = 2$ のときの行列積モデルを図 5 に表す．$(A')^{-1}$ の結果が各ノードに分散して配置されていることに注意して欲しい．いま，ステップ 1 からステップ 9 までを 1 ステージと定義する．Hybrid1 では，計算途中にデータ再配置が存在せず，この 1 ステージを高速に求解することは可能である．しかし，計算開始時点で各ノードが，A^{-1} の全体を保持する必要がある．そのため，2 ステージ目に移るとき，各ノードで求めた A^{-1} の部分行列の gather が全ノードで必要（Alltoall の通信）となる．このとき，1 ノードあたりの通信量は $O(n^2)$ となる．この通信量は非常に大きく，並列化に伴うメリットを失ってしまう．

3.2 Hybrid2

Hybrid2 は処理全体をデータ並列化するものである．Hybrid1 で必要であったステージ間でのデータ再配置を不要とするものである．計算開始時点で，各ノードは A^{-1} を行方向ブロック分割した部分行列を所持している．Hybrid1 と同様のアルゴリズムを用いることで，次のステージで必要となる A^{-1} の部分行列が各ノードに存在することとなり，ステージ間でのデータ再配置が不要となる，一方で，A^{-1} がブロック分割して分散配置されることに対応して，ステップ 1〜6 も A^{-1} の配置形態に対応したデータ並列処理の検討が必要となる．したがって，E_c は列方向のブロック分割を行い各ノードに配置するものとする．この際，ステップ 1〜6 を全てデータ並列処理するとステップ間でのデータ再配置が必要となりボトルネックの要因となる．そこで我々はステップ 1，2 および 5 のみをデータ並列処理，ステップ 3，4 および 6 を重複処理させる方法を採用し，さらにステップ 2 および 5 の処理を，各ノードで部分和（S_2' および S_5'）を計算する処理と，データ再配置を行う処理，部分和を合成する処理に分割し，データ再配置とその他の処理をオーバラップ実行することで通信時間を隠蔽するための計算順序の入れ替えを行う手法を提案した?．$p = 2$ のときの従来法，Hybrid1 および Hybrid2 の計算ステップの比較図を図 6 に示す．

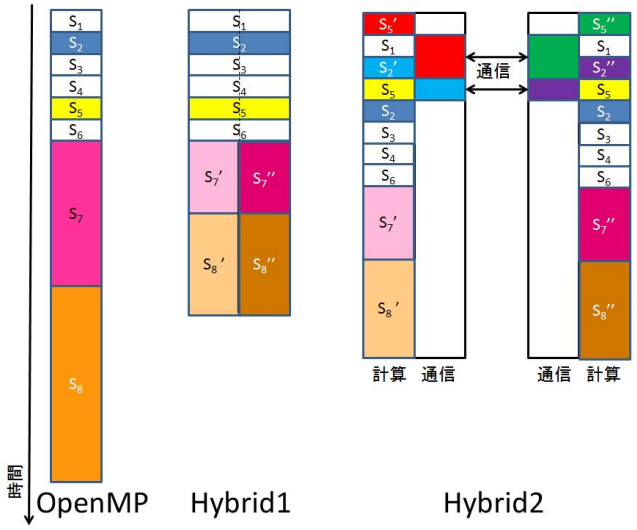


図 6 各手法のタイムステップの比較

表 1 並列度による計算時間と通信時間のモデル

	計算時間	通信時間
従来法	$O(sn^2)$	-
Hybrid1	$O(\frac{sn^2}{p})$	$O(n^2)$
Hybrid2	$O(\frac{sn^2}{p})$	$O(sn\log p)$

4. モデル化

SMW 公式の計算量は $O(sn^2)$ となることは既に述べた．Hybrid1 は図 3 の S_7 以降が並列化される，すなわち $O(sn^2)$ の部分が並列化されるため，p 台でノード間並列を行った場合，計算量としては以下になる．

$$O(\frac{sn^2}{p})$$

また，連続したシミュレーションを行う場合，データ再配置として $O(n^2)$ の時間がかかる．Hybrid2 に関しては，S_5 および S_7 以降が並列化される．したがって，計算量は Hybrid1 と同様となるため，

$$O(\frac{sn^2}{p})$$

となる．また，計算途中の通信として $O(sn\log p)$ の時間がかかる．これらの通信の時間を含めたモデルを表 1 に示す．

5. 評価

異なる行列データである Liver1253（京大医学部），nasa1824（Florida Matrix Collection），plat1919（Florida Matrix Collection）を用いて評価を行った．N の大きさは Liver が 3759，nasa が 5472，plat が 5757 である．S は 32 で固定とした．実行環境を表 2 に示す．ノード間並列度は 2 で固定とし，ノード内並列度を変化させ評価を行った．

表 2 実験環境

	環境 1	環境 2	環境 3
CPU	Core2Quad Q9550 2.83GHz L2Cache 12MB	Core i7 3770 3.4GHz L3Cache 8MB	Xeon X7560 2.26GHz×4 L3Cache 24MB
Memory 容量	2GB	8GB	128GB
Memory Band width	12.8GB/s	25.6GB/s	10.6GB/s
Network	Gigabit Ethernet	Gigabit Ethernet	10Gigabit Ethernet
OS	Linux2.6.191.2895.fc6	Linux2.6.35.14-106.fc14	Linux2.6.18-23.1.1.el5 x86_64
Compiler	mpiicc,icc12.0.0	mpiicc,icc12.0.0	mpiicc,icc12.1
Compiler Option	-O3	-O3	-O3
MPI	Intel MPI	Intel MPI	Intel MPI
Library	MKL10.0.3.020	MKL10.3.0.084	MKL 10.3 Update6

5.1 実行時間の評価

Liver1253 での結果を図??，nasa1824 での結果を図??，plat1919 での結果を図??に示す．環境 1 においては，全ての行列データで，Hybrid 並列は同じスレッド内並列度の従来法より約 2 倍の高速化が得られた．環境 2 においては，Hybrid1 にでは約 1.8 倍，Hybrid2 にでは約 1.5 倍の高速化が得られた．環境 3 においては，スレッド数 1 の場合においては約 1.8 倍の高速化が得られているが，並列度があがるにつれ，高速化の割合が減少している．

さらに，従来法と Hybrid の並列度が同じになるように着目すると，環境 1 では，おおよそ 1.4 倍，環境 2 ではおおよそ 1.2 倍の高速化が達成されていることがわかる．環境 2 は環境 1 よりもメモリバンド幅が広いため，高速化の割合が少なくなっている．しかしながら，環境 3 においては Hybrid 並列よりも従来法が高速である．これは，メモリバンド幅が他の環境に比べて広いため，この規模の行列ではボトルネックが生じていないことが考えられる．以上より，環境 1 および環境 2 では 3759 元以上の問題に対してハイブリッド並列は有効である．

また，環境 2 ではどの行列データを用いた場合でも Hybrid2 は Hybrid1 と比較して 10msec 程度遅くなっている．これは，環境 1 や環境 3 よりも CPU の性能が良いため，オーバーラップ実行を行っている部分 (S_1) の計算が高速であるため，オーバーラップ効果が薄まっているからであると考えられる．環境 3 で Hybrid2 が Hybrid1 と比較して高速である理由としては，他の計算機のネットワークの 10 倍の速度であるため，オーバーラップ効果が高いからである．しかしながら，並列度が上がるにつれ，オーバーラップ効果が薄まるため，最終的には Hybrid1 よりも遅くなっている．

最終的に得られた結果としては，Hybrid1 が Hybrid2 に比べ高速である．ただ一度だけ逆行列を求めるのであれば，Hybrid1 の方式が優れている．しかしながら，3.1 で述べたように，Hybrd1 の並列化手法では，新たなステージに移ったとき，すなわち，再度逆行列計算が必要となったときにデータの再配置が発生する．データ再配置にかかるコストは大きく，図??に示すように，ハイブリッド並列を行って得たメリットを失ってしまう．そのため，実時間シミュレーションへの応用には Hybrid2 の方式が最適である．

表 3 環境 1 におけるスレッド数が 1 のときの Liver の計算時間

	従来法	Hybrid1	Hybrid2
逆行列計算時間	294	149	154

5.2 計算時間と並列度とのモデルの評価

今回は並列度 $p = 2$ とし，ノード間の通信速度がピークの約 8 割と仮定し，評価を行った．スレッド数 1 の時を考える．表 3 を例に確認してみると，まず従来法での逆行列計算時間は 294msec となっている．この速度にモデルを適用すると，Hybrid1 および 2 は 147msec となる．これは，Hybrid1 の実測値 149msec とほぼ等しい．Hybrid2 は計算途中でデータ再配置があるので，そちらも考慮する．まず，S のサイズは 32 であり．N のサイズは 3759 である．また，S，N ともに double 型であるので，通信を行う大きさは 962,304Byte であり，8.4msec となる．したがって，Hybrid2 にかかる時間はおおよそ 157msec となり実測値 154msec とほぼ等しい．

6. まとめと今後の課題

本論文では実時間時系列シミュレーションの高速化を目的とし，まず最初に連立方程式 $Ax = b$ の求解問題に対して，係数行列 A の近似行列の逆行列が既知の場合に，SMW 公式を用いて A の逆行列を求めて高速に求解する手法について解説し，ハイブリッド並列化について議論した．その結果，環境 1 および環境 2 においては 3759 元の疎行列問題に対して，差分行列サイズ 32 に相当する係数行列の変化が起こった場合に，ハイブリッド並列処理を行うことでおおよそ 1.4 倍の高速化を達成した．また，並列度による計算時間と通信時間のモデル化を行い，それが正しいことを示した．今後，ノード間並列度の向上および並列度が

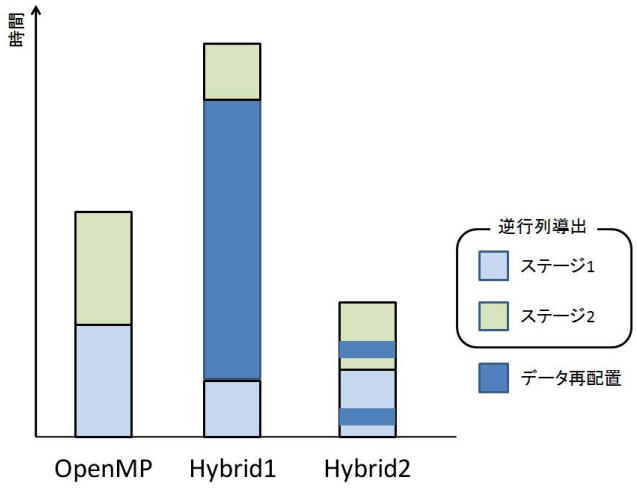


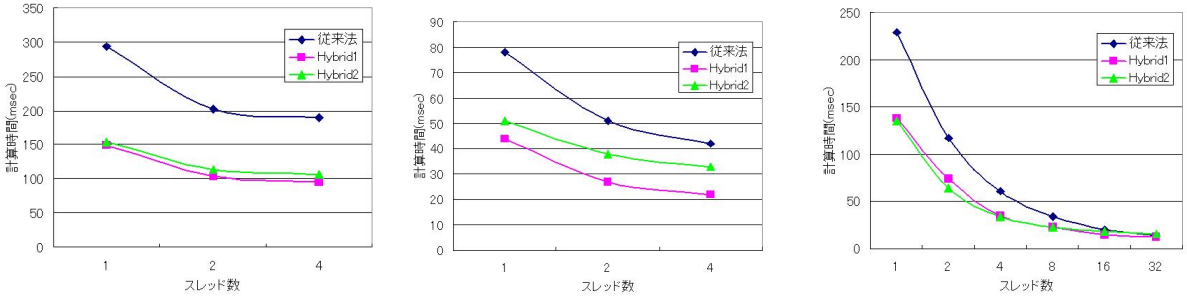
図 7 2 ステージ実行時の各手法の所要時間（概念図）

2 より大きい場合の計算時間と通信時間モデルの評価，環境 3 において，行列サイズがどの程度大きくなるとハイブリッド並列が有効であるかの検証を行っていく．

謝辞 本研究の一部は,JSPS 科研 基盤 (C)22500044 ならびに JST A-STEP(FS) 探索ステージ AS242Z00401H の助成による。本研究の実施にあたり、日頃より計算機環境の維持管理に貢献いただいている松山幸雄技術職員、ならびに、活発な議論を通して貴重な御意見を頂いている森・福間研究室の諸氏に心より感謝いたします。

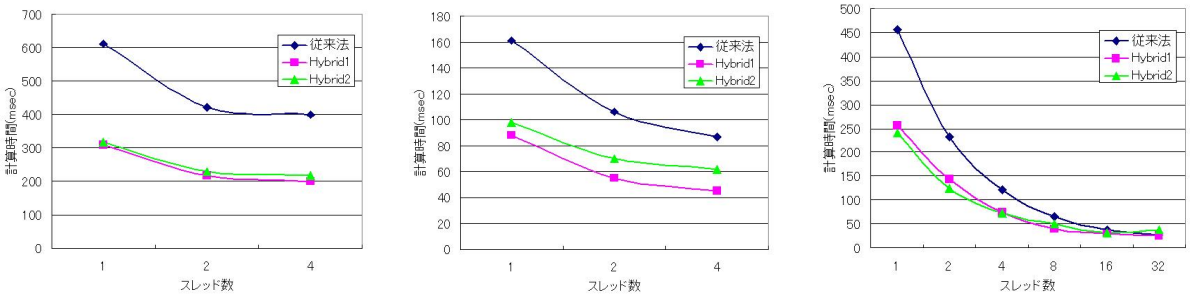
参考文献

[1] G. H. Golub and C. F. Van Loan, Matrix Computations(3rd Ed.), Johns Hopkins Univ. Press, 1996.
[2] 岩永翔太郎, 福間慎治, 森眞一郎, 実時間シミュレーションへの応用を前提とした SMW 公式を用いた逆行列計算のマルチコア並列処理, 信学論, Vol.J94-D, No.7, pp1165-1168, 2011 年 9 月
[3] Shotaro IWANAGA, Shinji FUKUMA, Shin-ichiro MORI, Hybrid Parallel Implementation of Inverse Matrix Computation by SMW Formula for Interactive Simulation, IEICE Trans. Info. & Sys., Vol.E95-D, No.12, pp2952-2953, Dec.2012
[4] 松井祐太, 福間慎治, 森眞一郎, 実時間シミュレーションへの応用を前提とした SMW 公式を用いた逆行列計算のハイブリッド並列処理の改良, 平成 24 年度電気関係学会北陸支部連合大会 講演論文集 (CD-ROM), F-13, 2012 年 9 月



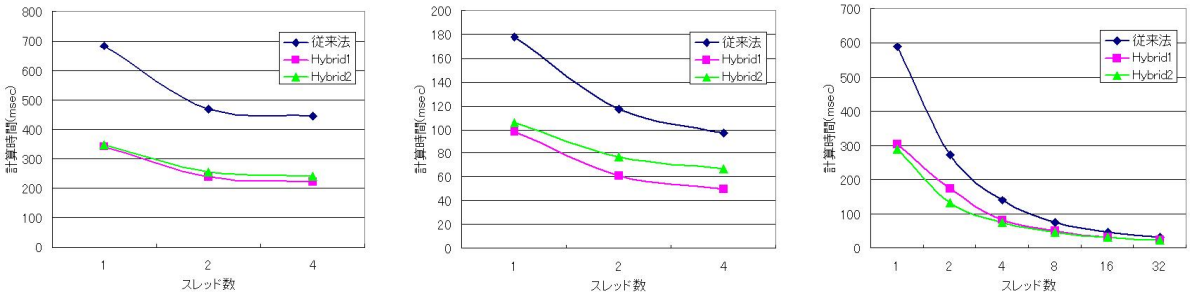
(a) 環境 1　(b) 環境 2　(c) 環境 3

図 8　行列データ Liver1253 の結果



(a) 環境 1　(b) 環境 2　(c) 環境 3

図 9　行列データ nasa1824 の結果



(a) 環境 1　(b) 環境 2　(c) 環境 3

図 10　行列データ plat1919 の結果