システムＬＳＩ設計技術　99－14
（2001．　1．12）

回路シミュレーションにおける並列分散デバイスモデル評価の実装

鈴木　毅，八木　浩行，檀　良

法政大学
〒184　東京都小金井市梶野町３－７－２
TEL：(0423) 87－6208　FAX：(0423) 87－6381
E-mail：tsuyoshi@dang.k.hosei.ac.jp

あらまし　我々は，回路シミュレーションにおける並列分散デバイスモデル評価を提案する．並列分散デバイスモデル評価は，デバイスモデル評価を並列化する事で，精度を維持したまま高速化することができる．本論文では，ダイオードモデルにおいて並列分散デバイスモデル評価を実装し,性能評価を行う．本手法を用いると，デバイスモデル評価時間は，従来の SPICE2 に比べ処理時間の 53.5~68.15%を低減でき，全体として最大で 1.7 倍の高速化が達成できる．

キーワード　デバイスモデル評価，回路シミュレーション，PC クラスタ，マルチスレッド

An Implementation of Parallel Distributed Device Model Evaluation on Circuit Simulation

Tsuyoshi Suzuki, Hiroyuki Yagi and Ryo Dang

Hosei University
3－7－2　Kajino-cho, Koganei-shi, Tokyo 184, JAPAN
TEL：(0423) 87－6208　FAX：(0423) 87－6381
E-mail：tsuyoshi@dang.k.hosei.ac.jp

Abstract　We propose a parallel distributed device model evaluation algorithm in the circuit simulation. Parallel distributed device model evaluation can speed up while maintaining accuracy, because the approach chosen to speed up SPICE is parallelizetion of device model evaluation. In this paper, we estimate the performance of this circuit simulator using the parallel distributed device model evaluation algorithm. The parallel distributed device model evaluation is able to reduce the device model evaluation to 53.5~68.15% in comparison with the original SPICE2. As a result, our circuit simulator can run 1.7 times faster than the original SPICE2 in the maximum speedup.

key words　circuit simulation, device model evaluation, cluster computing, multithread

1. はじめに.

半導体素子製造技術の飛躍的な発展によりトランジスタの微細化が進み,その結果, 集積回路の高集積化と共に設計技術も進歩している. 超大規模回路設計においてはディジタル回路においてもアナログ的回路の振る舞いが重要視されつつあり, 昨今のトレンドである System On Chips(SOC) 設計の核をなすシステムLSIにおけるデジアナ混在回路では, もはや全アナログ回路としての解析が必要とされ, 高速で SPICE レベルの精度を保つアナログ回路シミュレータに対するニーズは増すばかりである. 最近, ハイ・パフォーマンス・コンピューティングの分野において, PC クラスタという安価で高速な並列計算機の出現により, PC クラスタ上で利用可能な並列アルゴリズムの研究が各分野で行われている. 回路解析分野においてもデバイスモデル評価の高速化という観点からは, 従来より様々な提案が行われているが, モデルパラメータが物理的意味を持ち, 精度を落とす事なく, モデル計算時間を短縮できるアルゴリズムは, 我々の知る限り皆無である. またより安価な PC 環境での回路シミュレーションの高速化における要求も無視できない段階にはいっている.

本論文の主提案は, 回路シミュレーションにおける並列分散デバイスモデル評価アルゴリズムである. これは, 近年容易に構築できるようになった小規模 PC クラスタにターゲットを絞ったデバイスモデル高速化アルゴリズムである. 本論文の残りの章で, これについて説明を行う. 2 章では, 標準回路シミュレータにおける過渡解析の実行時間の分析を行い, 3 章で, 我々の提案する並列分散デバイスモデル評価アルゴリズムについて説明を行う. 4 章は, 実際にダイオードモデルにおいて並列分散デバイスモデル評価を実装した回路シミュレータを用い, 構築したPCクラスタにおいて性能評価を行う. 5 章で結論をまとめる.

2. 過渡解析の実行時間の分析

SPICE2[(1)]の過渡解析では, 各時間ステップにおいて容量 C やインダクタンス L などのエネルギー蓄積素子を離散化し, ニュートンループにおいて非線形素子を線形化して行列求解を行う. 従って, ニュートンループ反復が解析の最内ループとなり, ループ内ではモデル関数の計算, 回路行列のロード, 行列求解, 収束のテストなどが行われる. 図1にSPICE2とDanSpice(SPICE2ベースの回路シミュレータにコード生成法[(2)]を導入したもの)の過渡解析における各処理の実行時間の比率を示す. これは, サンプル回路として 1000~1 万段の Diode を鎖上に接続した回路と, 100~1000 段の ECL ゲートの縦続接続回路と n-mos の 2～64 ビット加算器回路の過渡解析を行い, それらの平均実行時間比率を示したものである. このデータによると, 行列求解のための時間は, SPICE2 で 28.7%, DanSpice で 19.5% である. これに対し, モデル関数の計算と行列要素のロードからなるデバイスモデル評価の処理は, SPICE2 で 60.2%, DanSpice で 68.2%を占めている. また SPICE2 では, デバイスモデル評価に全解析時間の平均 80%を要するという報告もある[(3)]. この事実は, SPICE2 において, スパース行列に対するコード生成法の効果により, 行列求解時間の占める比率は比較的小さい事を示している. また行列求解については, コード生成法だけでなく, スパースな回路行列に対して有効な LUCAS[(4)]などのハードウエアソフトウエア協調コンピューティングの特性を生かした高速化処理法も提案されている. 従って, SPICE に代表される標準回路シミュレーション法を高速化するためには, 計算コストの最も大きいデバイスモデル評価の処理について, 高速化を図る必要がある. デバイスモデル評価の高速化技法としては, テーブルルックアップ法[(5-7)]とバイパス法などが提案されている. しかし, テーブルルックアップ法は, 確かにある定数ファクターでのスピードアップを達成できるが, デバイス特性の回路動作への関係が直接見えにくいという問題がある. またバイパス法は, それを導入するとデバイスモデル評価をかなり軽減できるがその不完全さは周知の通りである. そこでモデルパラメータが物理的意味を持ち, モデル計算時間を短縮できる並列分散デバイスモデル評価アルゴリズムについて提案を行う.

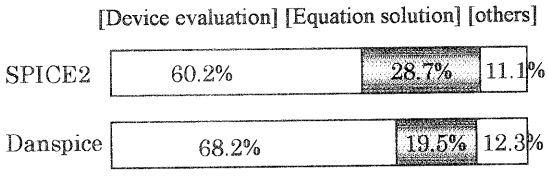

図1　過渡解析の各処理の実行時間比率

3. 並列分散デバイスモデル評価アルゴリズム

3.1 速度向上率の見積もり

図 2 に, アムダール法則により見積もった並列分散デバイスモデル評価の理想的な速度向上率を示す. このデータによると, SPICE2,Danspice 共に, 8 プロセッサまでは急

激な速度向上を見込めるが，それ以上プロセッサを増やした場合，速度向上率は飽和する．これは全解析時間においてデバイスモデル評価が占める割合がコード生成法を導入したDanspiceにおいても68.2%と小さいためで，スパースな回路行列に対して有効なLUCASを適用した場合，全解析に占めるデバイスモデル評価の割合が大きくなり，更なる速度向上も期待できるが，今回我々の提案は，あくまでこのデータに基づき，見積もりでよい速度向上率が得られる8プロセッサまでの小規模 PC クラスタシステムにおける並列分散デバイスモデル評価アルゴリズムの提案を行う．

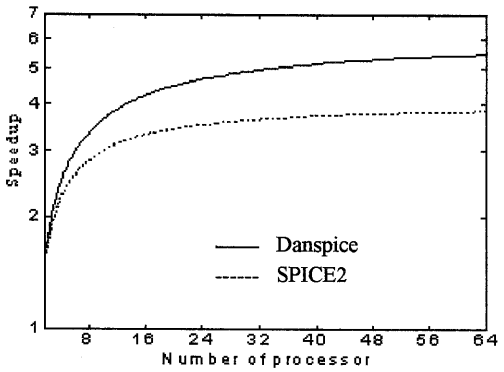

図 2　過渡解析における速度向上率の上限

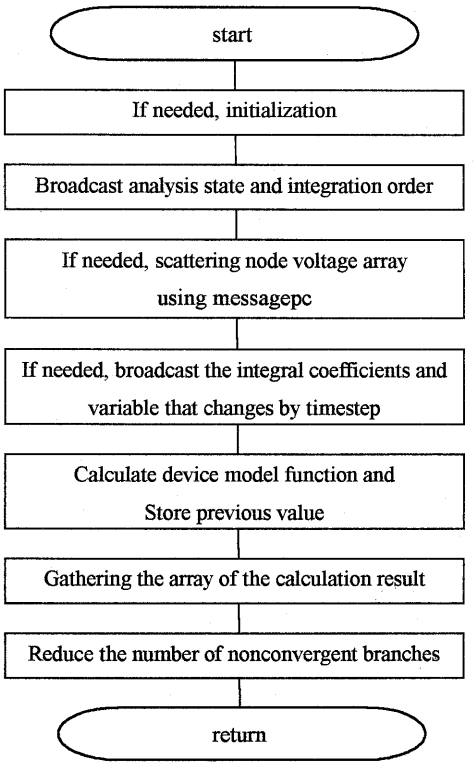

図 3　並列分散デバイスモデル評価の処理フロー

3.2　モデル計算部の並列化技法

SPICE2 のデバイスモデル評価を詳しく見ると，前処理部，初期値設定部，バイパス判定部，枝特性計算部，積分部，収束判定部，回路行列のロード部からなる．この内回路行列のロード部はルートノードで行わなければならないので次節で述べる．モデル計算部というとデバイスモデル評価において回路行列のロード部を除いた部分を指し，このモデル計算は，素子間の相互作用がなく，それぞれ独立に処理する事が可能である．図 3 にモデル計算部の並列処理フローを示す．ここでは，各イタレーションにおけるモデル計算の効率的な並列化技法について述べる．

(1)　Do 文によるループ

SPICE2 では，直流・過渡解析ルーチンにおける素子モデルのループは，図 4(a)のような goto 文によって構成されている．ここで locate(id)は素子の変数テーブルにおける最初のポインタを格納している．goto 文によるループのように，命令を実行する以前にループの回数が決定されないループはデータ並列処理できない．そこで図 4(b)のように回数を予め'jelcnt'に代入し，goto 文によるループを do 文によるループに変更している．また回路行列のロード部についても do 文によるループにするが，ルートノードのみの処理なので，モデル計算部の do ループとは分離し，モデル計算部のみ PC クラスタを用いたデータ並列処理を行う．

(a)　goto 文によるループ

```
      loc=locate(id)
10    if(loc.eq.0) return
          回路行列の計算
      loc=nodeplc(loc)
      go to 10
```

(b)　do 文によるループ

```
      do i=1,jelcnt(id)
          回路行列の計算
      enddo
```

図 4　do 文によるループ

(2) データ構造の再構成と動的記憶域管理法

SPICE2 では，動的記憶域管理法によってデータ領域を管理し，そのデータ領域を必要に応じて間接データ参照することで，計算を行う．そのため，過渡解析ルーチンでは，必要に応じてバイパス・打切り変数における各時刻のデータ領域を動かす．しかし，PC クラスタは，このままでは，ルートノード以外のノードにおいて，このデータ領域の移動は行われない．そのため，バイパス・打切り変数における各時刻のデータ領域の値に関しては，動的記憶域管理法に基づいたデータ領域を使用せずに，予め領域を確保しておいた配列に連続データとして保存し，initf=5,6(表 1 参照)の場合のみ，各ノードで連続データとなっている配列を予め領域を確保しておいた別の配列に移動し，それらを連続データ参照させる事でデータ領域を動かし，通信を回避する．

表 1 解析状態フラグ' INITF' の意味

INITF	意味
1	直流・過渡解析の解の収束
2	動作点解析の初回
3	動作点解析の 2 回目
5	過渡解析 時刻 t=0
6	過渡解析 各時刻での初回

(3) 通信の最適化

まず各イタレーションにおいて，解析状態をあらわす変数 initf と iord(積分オーダーのフラグ)を，MPI_BCAST によってルートノードからその他のノードに変数の送信を行う．図 5 に，通信を行うモジュール messagepc の木構造を示す．messagepc は，各ノードに送信された解析状態フラグに応じて，必要な変数や配列を送信する．TRAP とは台形公式(数値積分に通信の少ない台形公式のみ採用)を指し，その後に続く数値は解析状態フラグを指す．ここで TRAP2 は，データ構造の再構築や，MPI_SCATTERV と MPI_GATHERV で使う整数配列の計算などの初期化作業を行う．TRAP1 は，予め領域を確保しておいた配列に節点間の電圧を格納し，MPI_SCATTERV によってルートノードから各ノードに必要な配列の送信を行う．TRAP3 は，ダイオードのモデル文で OFF オプションが指定されていなければ TRAP1 と同様の作業を行い，指定されていた場合は何も行わない．TRAP5 は，数値積分に必要な積分係数のみ MPI_BCAST によってルートノードからその他のノードに変数の送信を行う．TRAP6 は，タイムステップにより変化する変数と積分係数を予め領域を確保しておいた作業用配列に格納し1つの配列にしてから，MPI_BCAST によってルートノードからその他のノードにこの配列を送信し，変数に配列をロードする．ここまでが，モデル計算前に必要なデータの分配である．モデル計算が終了した後には，MPI_GATHERV を用いて計算結果を収集する．まず予め領域を確保しておいた配列に計算結果を格納し，1つの配列にしてから送信している．最後に各ノードにおける未収束な素子の数を MPI_REDUCE により通信しながら合計を求める．

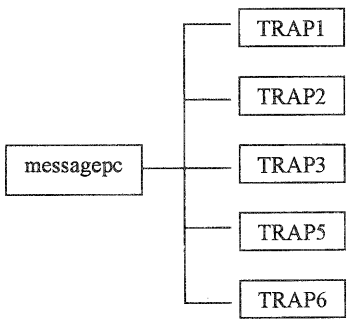

図 5 messagepc の木構造

3.3 回路行列のロード部における並列化技法

SPICE2 の回路行列要素は，ニュートン反復における各イタレーションにおいて，時間や応答変数の変化に応じてロードしなければならない．また右辺ベクトル要素も，外部の独立電源の他に，非線形依存電源の線形化及びリアクタンス素子の離散化などから生じる等価電源を含むため，各イタレーションにおいてロードしなければならない．これらの要素の値は，素子の接続方向も考慮した接続関係から決定される．ここでは，各イタレーションにおける行列要素および右辺ベクトル要素の効率的なロード方法について述べる．

まず 3.2 節で述べた do 文によるループにより，回路行列のロード部はモデル計算部の do ループから分離し，回路行列のロードのみを行う do ループとなる．この do ループも，モデル計算部と同様素子間の相互作用はなく，素子数だけ繰り返し計算を行う．モデル計算部と異なる点は，回路行列のロード部は，ルートノードの同じメモリ空間において処理されなければならない点である．次に対称型マルチプロ

セッサシステム(Symmetric MultiProcessor System, SMP)上において，マルチスレッドプログラミングによりデータ並列処理を行う．スレッドの場合，同じメモリ空間内で動作するため，回路行列のロード部におけるデータ並列処理はループを分割するだけで実現できる．これにより，回路行列のロード部においても効率よく並列化する事が可能となる．

4. 並列分散デバイスモデル評価の性能評価

ここまでに述べた並列分散デバイスモデル評価を非線形素子で最もデバイスモデル評価が小さいダイオードモデルに実装し，並列化の効果をテストした．

4.1 使用する計算機環境

テストで用いた PC-Cluster 型並列計算機は 4 台の SMP マシンから構成される．各ノードの接続には，FastEthernet を用いている．FastEthernet はスイッチングハブによって接続されている．表 2 に用いた PC-Cluster のスペックを示す．また並列プログラムの記述には MPICH[8] と呼ばれる通信ライブラリを用いて，メッセージ・パッシング型の並列プログラムを記述し，pthread ライブラリを用いてマルチスレッドプログラムを記述している．

表 2 用いた PC-Cluster の仕様

Processor	PentiumⅢ 500MHz*8
Memory	128MB*8 (Master は 192MB)
BCL	100Mbps FastEthernet, TCP/IP
OS	Linux2.2.12
通信ライブラリ	MPICH 1.2.0
Pthread ライブラリ	Linux thread 0.8(glib 2.1.2)
F77 コンパイラ	Pgf77 3.1[9]

4.2 テスト回路

今回はダイオードモデルの実装であるから，テスト回路として，多段ダイオードチェーン回路を使用した．

4.3 デバイスモデル評価のテスト結果

図 6 に，1000～5000 段のダイオードチェーン回路を実行した場合におけるデバイスモデル評価の速度向上率を示す．ここでは，回路行列のロード部については並列化を行っていない．それは，ダイオードの回路行列のロード部のみ並列化した場合，現状では逆に遅くなるからである．(これについては，次節で述べる)．そのため，図 6 は，モデル計算部のみ並列化を行った結果である．これにより，2～6 プロセッサにおいては，回路規模に応じて速度向上率が上昇している．しかし，それ以上プロセッサを増やした場合，6 プロセッサの場合よりも速度向上率は低下する．これは，6 プロセッサ以上では，通信時間が増加したためと考えられる．図 7 にこのテストにおける通信時間とデバイスモデル評価の実行時間を示す．これにより，デバイスモデル評価は，プロセッサを増すほど実行時間が減少し，一方通信時間は，プロセッサを増やすほど増加している事が解る．結果として，この場合，プロセッサを増やして各ノードにおけるモデル計算部の負荷を減少させたとしても，それよりも通信時間が増えてしまう事で，デバイスモデル評価全体として速度向上率が低下する．図 8 に，5000 段のダイオードチェーン回路を実行した場合の通信時間の詳細を示す．これにより，モデル計算終了後に，各ノードの計算結果を収集する MPI_GATHERV という関数のみ非常にコストが高い事が解る．MPI_GATHERV は，各ノードに分散した計算結果をルートノードに収集するため，デバイスモデル評価を並列化した場合の通信において最もコストの高い通信である．これは 2 つの理由によるものである．1 つは収集する要素が回路規模に応じて，(各ノードにおいて割り振られた素子数)×2 のオーダーで増加するからである．もう一つは MPI_GATHERV が，モデル計算終了後，一括して各ノードが持っている計算結果が格納された配列を収集するための通信であるからである．つまり，MPI_GATHERV の場合，通信そのものの時間と通信における待ち時間が，プロセッサを増やせば増やすほど他に比べて増加するという事になる．このため，モデル計算部を並列処理した場合，現状のネットワーク性能では，並列化した際，通信時間がかなりのオーバヘッドとなり，高速化を妨げる．

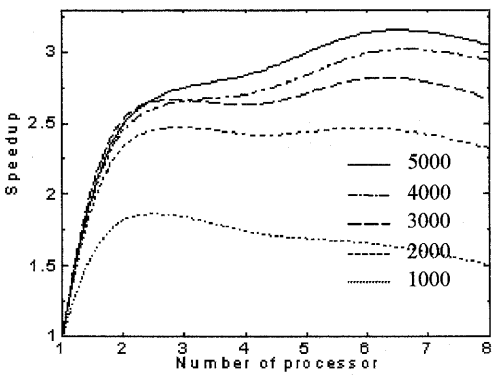

図 6 速度向上率

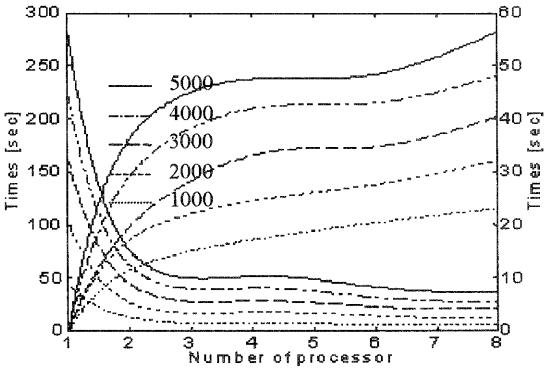

図 7 通信時間とデバイスモデル評価の実行時間

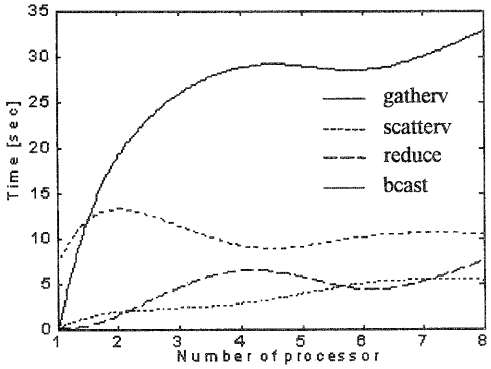

図 8 通信の分析

4.4 回路行列のロード部のテスト結果

図 9 に, 1000～6000 段のダイオードチェーン回路を, 3 つの場合において実行した時の, 回路行列ロード部における実行時間を示す. 1つ目は, 回路行列のロード部がgoto 文のループで構成されるオリジナルの SPICE2 である. 2 つ目は, 回路行列のロード部を do ループに変更した SPICE2 である. 3 つ目は, 回路行列のロード部をマルチスレッドにより並列化した回路シミュレータである. これにより, 回路行列のロード部を並列化した場合, 1 つ目におけるオリジナルの SPICE2 よりは高速であるが, 2 つ目の do ループ化したものに比べて, 遅くなることが解る. これには 2 つの原因が考えられる. 1 つは, メモリのバンド幅に起因するものであると考えられる. この事実を示すため, フリーソフトウエア lmbench[10]を使用して, 複数のスレッドを起動した際のメモリリード, メモリライト, bcopy のバンド幅を計測した. 結果を表 3 に示す. 示されているバンド幅は全スレッドのバンド幅の合計である. 2 スレッドで行う場合, メモリのリードは 1.56 倍, メモリのライトは 1.23 倍と 2 スレッドではバンド幅が低くなるため, ほとんどメモリのリードライトしか行わない回路行列のロード部では, このメモリバンド幅が影響し, 高速化の妨げになっていると考えられる. 2 つ目は, 現在の実装では, 各イタレーションにおいて, スレッドの生成, 消滅を行うため, 生成と消滅にかかるオーバーヘッドが高速化の妨げとなっていると考えられる. これら2つのオーバーヘッドは, 回路規模に関係なく一定である. そのため回路規模が大きくなるにつれて, 並列化の効果が現れる. 図 10 に, do ループ化した場合と並列化した場合の比率を示す. これは, 回路規模が大きくなるにつれて, より並列化した場合の効果が現れることを示している. よって, 大規模規模な回路においては, 高速化が見込める. また今回は, 回路行列のロード部の並列化を, ダイオードモデルにのみ実装したが, 回路行列のロード部は, 回路中に含まれる全デバイスモデルにおいて一括して行えるため, その部分を並列化することで更に並列化の効果が現れると考えられる. 参考のために表 4 に, 各デバイスモデルにおける 1 素子あたりの回路行列のロード数を示す. これは, 1 素子あたりの回路行列のロード数が MOS に関しては, ダイオードモデルの場合の 2.89 倍, BJT に関しては 3.1 倍とロード数が多い事を示している. よってより大規模で複雑な回路においては, 回路中に含まれる全デバイスモデルにおける回路行列のロード部を一括して並列化する事で, 高速化が見込める.

表 3 メモリバスのバンド幅

スレッド数	Bcopy [MB/s]	ratio	read [MB/s]	ratio	write [MB/s]	ratio
1	144.5	1	336.5	1	184.7	1
2	153.7	1.06	525.7	1.56	228.1	1.23

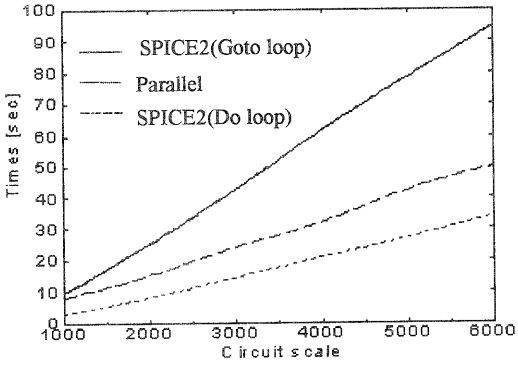

図 9 回路行列のロード部の実行時間

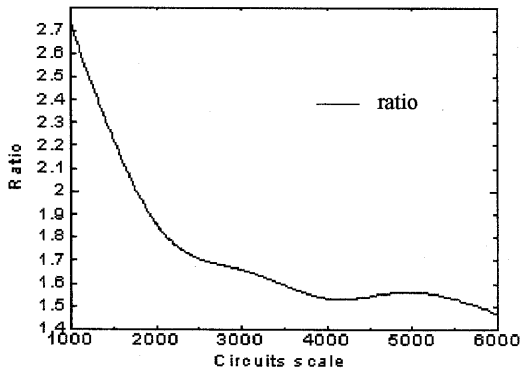

図 10 回路行列のロード部の実行時間比率

表 4 各デバイスモデルにおける回路行列のロード数

Model	Number of the element
resistor	5
capacitor	6
inductor	4
Nlvccs	8
Nlsvcvs	6
Nlcccs	9
Nlccvs	8
voltage sources	5
current sources	2
Diode	9
Bjt	28
Mosfet	26
Jfet	18
Tsline	22
Total	156

* nlvccs: nonlinear voltage-controlled current sources
nlvcvs: nonlinear voltage-controlled voltage sources
nlcccs: nonlinear current-controlled current sources
nlccvs: nonlinear current controlled voltage sources

5. まとめ

本論文では，標準回路シミュレータの高速化技法として，並列分散デバイスモデル評価アルゴリズムの提案を行い，構築した PC クラスタを用いて，ダイオードモデルにおいて実装した回路シミュレータにおいて，そのアルゴリズムの性能評価を行った．この提案により，デバイスモデル評価は，ダイオードモデルにおいてであるが，本手法を用いると，デバイスモデル評価は，1 回の評価あたり，従来の SPICE2 に比べ処理時間の 53.5~68.15%を低減でき，全体として最大で 1.7 倍の高速化が達成できる．しかしながら，通信時間のオーバヘッドが高速化の妨げとなっている事は明らかであり，今後の課題である．

参考文献

[1] L W. Nagel, “SPICE2: A computer program to simulate semiconductor circuits, ” Memorandum No.UCB/ERL M520, 1975.

[2] 八木浩行，檀 良，”PC 環境での回路シミュレーション用実行時コード生成法” 電子情報通信学会論文誌 A Vol.J-83A No.4 pp.444-446 2000 年 4 月．

[3] A. R. Newton and D. O. Pederson “Analysis time, accuracy and memory requirement tradeoffs in SPICE2” ,Proc. IEEE Int. Symp. Circuits & Syst, pp.6-9(May 1978).

[4] 八木浩行，檀 良，”LUCAS を使用した回路シミュレータの性能評価” 情報処理学会論文誌 Vol.41 No.4 pp.915-926 2000 年 4 月．

[5] T. Shima, T. Sugawara, S. Moriyama, and H. Yamada, “Three-Dimensional Table Lookup MOSFET Model for Precise Circuit Simulation,” IEEE Journal of Solid-State Circuits, Vol.SC-17, pp.449-453, June. 1982.

[6] P. Subramaniam, “Modeling MOS VLSI Circuits for Transient Analysis,” IEEE Journal of Solid-State Circuits, Vol.SC-21, pp.276-285, April. 1986.

[7] A. Rofougaran and A. A. Abibi,”A Table Lookup FET Model for Accurate Analog Circuit Simulation” IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems, Vol.12, no.2, pp.324-335, February.1993.

[8] http://www.mcs.anl.gov/mpi/index.html

[9] http://www.softek.co.jp/SPG/index.html

[10] http://www.bitmover.com/lm/lmbench/lmbench.html