SDNNアルゴリズムを用いた論理回路設計と故障診断の実験

2E−5

新井 正敏　中川 徹　北川 一
(豊田工業大学)

1. はじめに

近年, 集積回路が高集積化, 複雑化してきている. このため, 集積回路に対するテストパターン生成が非常に困難になってきている. また, 本学において, 集合論にもとづいたSDNN(Strictly Digital Neural Networks)による論理素子NLG(Neural Logic Gates)の提案がなされている. [1]

以下では, NLGによるテストパターン生成を行うことを目的とし, 実験を試みた.

2. 集合によるNLGのプログラミング

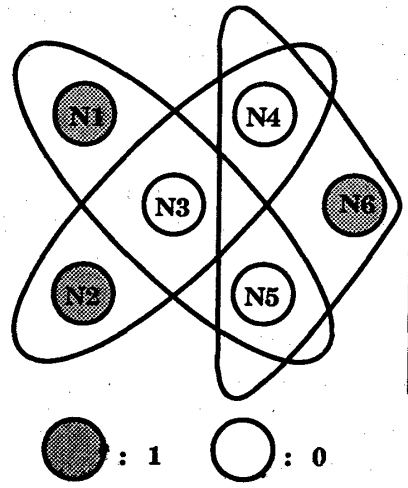
SDNNでは, 問題を複数の集合によって表現し, 集合内の要素n個からk個を選択するk-out-of-n設計規則にもとづいて, プログラミングを行うものである.

使用したNLGは, 図1に示すように, 6個のニューロンに3種の1-out-of-3制約条件を付けたものである. それぞれのニューロンにN1 ～ N6までの名前を割り当て, N1, N2を入力とし, N3, N6を出力とする. 入出力関係を真理値表に表すと

$$N3 = \overline{N1 + N2}$$

$$N6 = \overline{N1 \oplus N2}$$

になる.



truth table

N1	N2	N3	N6
0	0	1	1
0	1	0	0
1	0	0	0
1	1	0	1

図1 NLGのプログラミングとその真理値表

図1の網掛け部はON, OFF状態が制約条件を満足し, 入力がともに1である時の収斂状態を示している.

ここで, NLGについての特徴を述べておく.

- 入力, 出力において双方向性がある.
  出力を固定し, 入力を算出(連想)することも可能である.
- 従来の論理回路では構成できない回路も構成できる.
  5.で述べるように, 論理回路の出力端子間をNOT回路で結ぶことも可能である.

このNLGを組み合わせることによって, 以降の実験を行った.

3. 全加算器の収斂実験方法

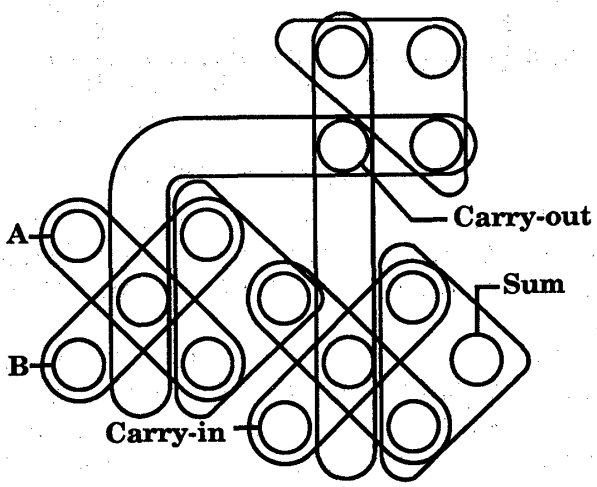


図2 NLGによる1ビット全加算器

NLGを用いた回路が収斂することを調べるために図2のような全加算器を構成し, 実験を行った.

規模が大きくなった場合の状況を知るために, 図2の1ビット全加算器をはしご状に並べて, 前段のCarry-outと次段のCarry-inが同一ニューロンになるように集合関係をプログラミングした.

本実験では, carry look-aheadなしでプログラミングし, 出力を固定して, 入力を連想するようにした.

A Logic Testing and Design for Testability using A Strictly Digital Neural Network
Masatoshi ARAI, Tohru NAKAGAWA, and Hajime KITAGAWA
Toyota Technological Institute

4. 実験結果と考察

Sun-3 上で Fortran にて記述された SDNN シミュレータを使用し，並列収斂ステップ数の平均値 ($\widetilde{Tp}$) と，1 つの解を求めるのに要する平均 CPU 時間 ($\widetilde{Ts}$) を測定した．

実測の結果，$\widetilde{Tp}$ のオーダは O(1) で，約 25 ステップであった．また，図 3 より $\widetilde{Ts}$ のオーダは O(n) であった．したがって，$\widetilde{Tp}$ とニューロン数の積のオーダが $\widetilde{Ts}$ のオーダと等しくなることが確認できた．

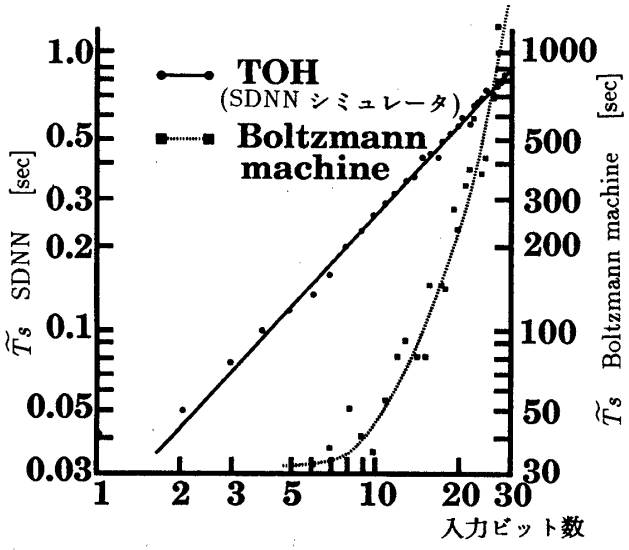


図 3 全加算器の収斂実験結果

さらに，他のニューラルネットアルゴリズム [2] との比較を行うために，C 言語にてボルツマンマシン [4] のシミュレータを作成し，$\widetilde{Ts}$ の比較を行った結果が，図 3 である．

ボルツマンマシンは，$\widetilde{Ts}$ がオーダ $O(10^n)$ であることがわかり，SDNN とは桁違いに時間がかかることがわかった．その理由は，SDNN ではニューロンの出力から入力が一意的に定まる決定的 (deterministic) なモデルであるのに対して，ボルツマンマシンは確率的 (stochastic) なモデルを使用しているからである．

したがって，今後の実験を進めるに当たり，SDNN を使用すればシミュレーションでも，問題規模を大きくした場合，比較的小さい $\widetilde{Ts}$ で計算できることが期待される．また，将来的に SDNN がハードウェア化された場合は前述の結果から O(1) で収斂することになるため，以下の実験では SDNN を使用していくことに決めた．

5. NLG によるテストパターンの生成方法

Chakradhar らは，テストパターンを求めるために，図 4 のような回路を提案している．[2]

図 4 は，正常な回路と異常を含む回路を出力側で NOT 接続したもので，図 4 が収斂した場合の入力と出力がテストパターンのペアである．

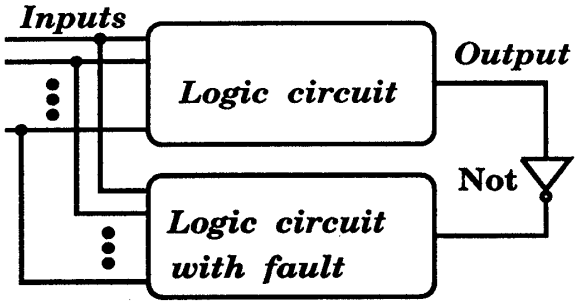


図 4 テストパターン生成回路

図 4 を NLG で構成することを考え，特に異常を含む回路のプログラム方法とテストパターン生成について述べる．ただし，今回取り上げる異常を含む回路とは 0 縮退故障と，1 縮退故障の回路を対象にしている．

- 異常を含む回路の構成
  正常な回路の一部を 0(0 縮退故障時) または，1(1 縮退故障時) に固定する．
- 入力の結合
  正常な回路と，異常を含む回路の入力端子を結合させる．
- 出力の結合
  出力端子間を NOT で結合する．さらに出力が複数ある場合には，対応する出力端子間を EXOR で結合し，それらの EXOR 出力を OR し，その出力が 1 になるようにする．

結果:

現時点で，全加算器のテストパターンを NLG で算出できることを確認できた．しかし，入力ビット数が増加するにつれ，$\widetilde{Tp}$ が O(n) で増加しており，この原因については調査中である．

6. おわりに

今回の実験は，規模として実用レベルにまでには達していないが，新しい考え方の一つとして進めて行く価値はあると思われる．

参考文献

1. 本大会別稿参照：中川他 "SDNN:・・・" 文献 [1]
2. S. T. Chakradhar, M. L. Bushnell, and V. D. Agrawal. : "Automatic Test Generation Using Neural Networks," Proc. ICCAD'88, pp.416-419, 1988.
3. H. Fujiwara. : "Logic Testing and Design for Testability," MIT Press, March 1985.
4. G. E. Hinton, T. J. Sejnowski, and D. H. Ackley. : "Boltzmann Machines : Constraint Satisfaction Networks that Learn," Tech Report CMU-CS-84-119, Carnegie-Mellon U.,May 1984.