3U-1

正則表現論理シミュレータの実現について

木村 晋二　崎山 和彦　羽根田 博正
(神戸大学・工学部)

1. まえがき

論理回路の大規模化とともに、論理設計の形式的検証手法に関する要求が高まりつつある。大規模な論理回路の検証を一つのレベルで行うことは計算量の点で問題があるので、論理検証を種々のレベルに分けて行うことのできる検証手法が重要である[1],[2]。ここでは、そのような検証手法の一つである正則表現論理シミュレーション手法の実現について報告する。正則表現論理シミュレーション手法は、論理回路を有限オートマトンとしてモデル化し、入力系列の正則集合に対して回路のシミュレーションを行う手法である。論理回路は、状態や入力記号の抽象度を変えることにより、低いレベルから高いレベルまで有限オートマトンでモデル化できるので、論理回路の種々の階層と正則表現論理シミュレーション手法の整合性は高い。今回実現したのは、ゲートレベルのシミュレーション部分である。実験により、シミュレータで扱う有限オートマトンの状態遷移の枝の数に比例した時間でシミュレーションできることが確認された。

2. 正則表現論理シミュレーション手法

ここでは、ゲートレベルの正則表現論理シミュレーションアルゴリズムについて述べる。このアルゴリズムは、組み合わせ論理回路の各ゲートの出力端子に現れる系列集合を受理する有限オートマトンを求めるものである。以下では、この有限オートマトンをその端子に対応する有限オートマトンと呼ぶ。なお、フィードバックループを持つ論理回路に対しては、フィードバックループを切断し、組み合わせ論理回路としてシミュレーションできることがわかっているので[3]、その場合でも本アルゴリズムが適用できる。

有限オートマトンのデータは、図1に示すようなリスト構造で表わす。入力記号は、図1に示すように配列型のものを用いる。配列の最初の部分は外部入力の値を、また最後の要素はその有限オートマトンに対応する端子の値を表わす。これは、シミュレーションの結果として、外部入力に対するその端子の値(入出力関係)が必要とされるからである。以下では、外部入力の部分を遷移(枝)の入力、端子の値の部分を遷移(枝)の出力と呼ぶ。

[シミュレーションアルゴリズム]

ここでは、論理回路の素子のシミュレーションの順序が決定されているとして、各論理素子に対する処理を示す。順序の決定には、通常のレベル付け[4]を用いればよい。なお、以下では2入力素子の場合を示す。

Step 1 素子の各入力端子に対応する二つの有限オートマトンfa_1とfa_2から、素子の論理演算を行った結果の有限オートマトンを構成する。これは、直積オートマトンを構成することに等しい。

1.1 fa_1, fa_2の各初期状態q_{01}, q_{02}を、初期状態対(q_{01}, q_{02})とし、これに対応するデータを割り当てる。

1.2 各状態対(q_1, q_2)に対し、以下の処理を行う。

1.2.1 q_1から射出する状態遷移の各枝eg_1に対し、以下の処理を行う。

1.2.1.1 q_2から射出しeg_1と同じ入力を持つ各遷移枝eg_2に対し以下の処理を行う。

(a) 各枝の遷移先の状態の対(q_1', q_2')が登録されているかどうか判定する。登録されていなければ新たに登録し、その状態対に対応するデータを割り当てる。

(b) (q_1, q_2)に対応する状態から(q_1', q_2')に対応する状態へ状態遷移の枝を付ける。入力はeg_1と同じとする。また、出力は、eg_1の出力o_1と、eg_2の出力o_2を論理演算した結果とする。

Step 2 Step 1で求めた有限オートマトンfaに対し、遅延操作を行う。これも基本的には、直積オートマトンを構成する操作である。遅延は純粋遅延とし、遅延時間をnとする。

2.1 faの初期状態q_0とX(unknown)のn個の並びの対を初期状態対とする。Xのn個の並びは遅延操作を行う有限オートマトンの初期状態に対応する。

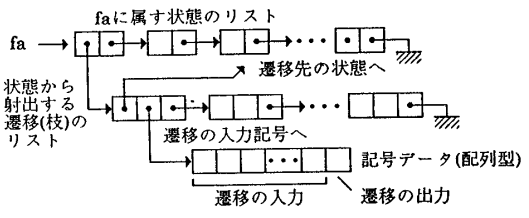

図1. 有限オートマトンのデータ

On the Implementation of a Regular-Expression Logic Simulator
Shinji KIMURA, Kazuhiko SAKIYAMA, Hiromasa HANEDA
Kobe University

2.2 各状態対$(q_1, (v_1, v_2, ..., v_n))$に対し、以下の処理を行う。

2.2.1 q_1から射出する各遷移枝eg_1に対し、以下の処理を行う。なお、q_1からの遷移のときの入力をin, 出力をo、遷移先をq_1'とする。

(a) $(q_1', (v_2, v_3, ..., v_{n-1}, o))$が登録されているかどうか判定する。登録されていなければ新たに登録し、その状態対に対応するデータを割り当てる。

(b) $(q_1, (v_1, v_2, ..., v_n))$に対応する状態から$(q_1', (v_1, v_2, ..., v_{n-1}, o))$に対応する状態へ状態遷移の枝を付ける。遷移の入力はin、出力はv_1とする。

3. 実現と評価

2.で示したアルゴリズムをFortran 77で実現し、その性能を評価した。ソースプログラムの行数は約700行である。評価に用いた回路は、(1) 2入力NANDゲートの入力を結合し、NOTゲートとしたもの一つからなる回路(ゲートの遅延時間を変化)、(2) 2入力で遅延時間1のNANDゲートの入力を結合し、NOTゲートとしたものを複数個結合した回路、(3) SR, T, D, JKの各フリップ・フロップである。なお、評価には、SONYのNEWS NWS-831を用いた。

まず、(1)の回路のシミュレーション時間および処理した有限オートマトンの個数、状態数、状態遷移の枝数を表1に示す。入力は(0+1)*としたので、出力の系列集合を表わす有限オートマトンの状態数は、遅延時間に対して指数のオーダーで増える。表1より、シミュレーションの計算量が、処理した枝数に比例していることがわかる。

つぎに、(2)の回路のシミュレーション時間および処理した有限オートマトンの個数、状態数、状態遷移の枝数を表2に示す。入力は(1)と同様(0+1)*としたので、扱う状態数は、遅延時間に対して指数のオーダーで増える。この場合も、シミュレーションの計算量は、処理した枝数に比例している。

最後に、種々のフリップ・フロップ回路のシミュレーション時間および処理した有限オートマトンの個数、状態数、状態遷移の枝数を表3に示す。表3には、回路の外部入力数、ゲート数、回路への入力系列集合を受理する有限オートマトンの状態数と枝数もあわせて示す。この場合も、シミュレーションの計算量は、処理した枝数に比例しているが、(1), (2)の場合に比べると、枝当たりの計算量は大きくなっている。これは、外部入力の数が、(1), (2)の場合より大きいことによる。

4. むすび

以上、正則表現論理シミュレーション手法の実現と評価について述べた。正則表現論理シミュレーション手法は、シミュレーションの入力として、時系列の正則集合を扱うものである。本手法は、論理回路の種々のレベルを扱うことができ、設計検証の道具として有用である。今回の試作で、シミュレーションに要する時間が、処理する状態遷移の枝の数に比例することが確認された。今後は、開発したゲートレベルのシミュレータをもとに、異なるレベルのシミュレータの実現などを考えてゆきたい。

謝辞 日頃から御討論頂く京都大学矢島脩三教授、神戸大学太田有三助教授ならびに神戸大学羽根田研究室の皆様に感謝します。

参考文献

(1) S. Gosh : "A Distributed Approach to Timing Verification of Synchronous and Asynchronous Digital Systems," IEEE Trans. on CAD, vol. CAD-6, No. 4, July 1987.

(2) K. J. Supowit and S. J. Friedman : "A New Method for Verifying Sequential Circuits," Proc. 23rd DA Conf., pp.200-207, 1986.

(3) 木村、羽根田 : "系列集合論理シミュレーション手法に基づく非同期式順序回路の検証," 信学技報 VLD87-118, 1988年2月.

(4) M. A. Breuer and A. D. Friedman : Diagnosis & Reliable Design of Digital Systems, Computer Scince Press, 1976.

表1 遅延時間の異なるNANDゲートのシミュレーション

遅延時間	CPU時間 (sec)	FA数	状態数	枝数
1	0.2	6	7	14
2	0.2	7	14	28
3	0.3	8	29	58
4	0.4	9	60	120
5	0.5	10	123	246
6	0.8	11	250	500
7	1.5	12	505	1010
8	2.8	13	1016	2032
9	5.5	14	2039	4078
10	10.7	15	4086	8172

表2 NANDゲートの直列接続回路のシミュレーション

接続数	CPU時間 (sec)	FA数	状態数	枝数
1	0.2	6	7	14
2	0.2	9	17	34
3	0.3	12	39	78
4	0.5	15	85	170
5	0.7	18	179	358
6	1.3	21	369	738
7	2.4	24	751	1052
8	4.7	27	1517	3033
9	9.2	30	3051	6102
10	18.3	33	6122	12242

表3 種々のフリップ・フロップのシミュレーション

種類	CPU時間 (sec)	FA数	状態数	枝数	外部入力数	ゲート数	入力FA状態;枝
SR F.F.	0.4	10	51	83	2	2	6;10
T F.F.	1.5	30	335	432	3	7	14;17
M-S T F.F.	12.4	43	1496	2616	4	11	37;53
D F.F.	1.0	18	136	277	3	4	8;13
JK F.F.	8.2	44	1828	2414	4	10	40;49