NEW APPROACH TO THE STATE REDUCTION IN INCOMPLETELY SPECIFIED SEQUENTIAL MACHINES

María J. Avedillo, J.M. Quintana and J.L. Huertas
Dpto. de Electrónica y Electromagnetismo, Universidad de Sevilla
Dpto. de Diseño de Circuitos Analógicos,
Centro Nacional de Microelectrónica, 41012 Sevilla, Spain

Abstract: We present ARNES, an algorithm for the state reduction of incompletely specified sequential machines. The new heuristic method, unlike other reported approaches, does not need to generate any complete set of compatibles. Starting from the set of internal states in the given symbolic description of the FSM, the application of a sequence of transformations results in a description with a smaller number of states. We also include experimental results over a wide set of machines which prove the superiority of the new algorithm.

Introduction

Systems for the design of Finite State Machines (FSM) have been implemented since the early 60's. Some of these systems have been used in industrial applications for more than 10 years but, few of them include state minimization and state assignment because the inherent complexity of these processes. In particular, it was shown [1] that the reduction of completely specified finite automata can be achieved in $O(nlogn)$ steps. The minimization of incompletely specified finite automata is a NP-complete problem [2].

Nowadays, technological advances lead to more and more sophisticated digital systems. In particular, more and more complex Control Units (CUs) are needed. It is impracticable to realize them without the help of CAD tools. Some automatic synthesis systems have been recently reported [3-5] to produce CUs implemented by FSMs, but no attention has been paid to the state reduction so leading the use of FSM with a great number of redundant states [6].

The minimization of the number of states is an important task in optimal design of sequential circuits. Reducing the number of states corresponds to decreasing the number of transitions of the sequencing functions (and eventually, to reducing the number of implicants in a two level logic realization). Moreover, a reduction of the number of states may correspond to a reduction of the number of bits that is needed for the state coding (simplified transition functions in a two level logic implementation) [7].

Another area where state minimization applies is the test generation for sequential machines. Extensions to the classical D-algorithm [8] or approximations based on random techniques [9] are ineffective when the number of states in the circuit is large and the tests demand long input sequences [10].

Classical methods begin by generating prime compatibility sets (PC sets) or any other set of compatibility classes for which a minimum cardinality closed cover composed uniquely of them exists. The number of these compatibility classes can grow very quickly with the number of internal states of the original machine description. Moreover, given a set of compatibles, the selection of a closed cover with minimum cardinality is also a NP-complete problem [2]. Heuristic approaches to the problem aim at getting near-minimum (minimal) cardinality solutions.

Heuristic Approaches to State Reduction

The classic heuristic method for state reduction is [11]. The author following the traditional structure of the problem solution, proposes a maximal compatible (MC) based approach, although there is no assurance of the existence of a minimum cardinality closed cover set composed uniquely of them. The use of this subset of the prime compatible set reduces (in some cases drastically) the number of compatible candidates. The procedure consists of selecting one of the essential (or quasi-essential) MCs and attempting to satisfy its closure requirements (generating one of the smallest set of MCs that satisfies the violated closure requirements for the MC selected). The result will be a closed set of MCs that may be or may not provide full cover on the initial set of states. The procedure is repeated until a full cover is performed.

REDUCES [12], like Bennets' algorithm, starts with the MCs as candidates to belong to a closed cover. The compatibles are selected according to a set of heuristic rules which select that M_i, which satisfies at least one of the most restrictive constraints (closure constraints more restrictive than cover ones and among each type those satisfied by a smaller number of MCs are more restrictive), and maximizes C a weighted sum of a) the number of violated constraints satisfied by selecting M_i (positive contribution)and b) the number of them that are violated as a consequence of adding M_i to the *set* (negative contribution). The procedure finishes when the selected set of MCs is a closed cover for the given FSM.

We can conclude that a common strategy underlies in both approaches we have briefly described. It consists in generating sets of compatibles (which can be computationally expensive and even prohibitive) and building, incrementally and heuristically, a subset of the previous collection. This subset is a closed cover of the FSM.

Description of New Algorithm

The new state reduction algorithm we propose differs in concept from those approaches described in previous Section. The strategy above, generation of compatibles and incremental building of a solution is abandoned in favor of one similar to that in ESPRESSO-IIC [13] for the heuristic minimization of combinational functions. This is, it is defined a set of basic operations which transform a symbolic description of the FSM in another one with a smaller number of states. The algorithm may be described, for a high level point of view, as a sequence of transformations (functions) which starting with the initial description of the FSM, results in a succession of intermediate descriptions of such machine with a decreasing number of states. The process finishes when the application of the functions implemented by the algorithm does no longer reduce the number of states. It main goal is that it does not need to generate any complete set of compatibles.

Primary objective in ARNES is minimizing the number of states in the symbolic description of a FSM being used as an input to other phases of the design process of sequential circuits. Moreover, ARNES, once it obtains a solution with a reduced number of states, operates on it in order to maximize the number of don't cares in excitation functions.

In Figure 1 the algorithm control block is described using Pidgin C.

```
ARNES(T)
   /* C   closed cover of state table T,
          initially the set of internal states in the
          symbolic description.
          Φ  cost function. Number of compatibles in
             C at each time.
          Initially number of states in table T */
   {  C = inicializa(T);
      Φ* = Φ = coste(C);
      (Φ,C) = expand(C,T);
      (Φ,C) = irredundant-cover(C,T);
      (Φ,C) = reduction(C,T);
      T' = table(C,T);
      if(Φ < Φ*)
      {  ARNES(T');
      }
   }
```

Figure 1: Description Pidgin – C of control block of the algorithm

Where:

T = symbolic description of a FSM using state tables.

C = closed cover for table T. This is, closed set of compatibles which cover all internal states in T.

Φ = cost function. Number of compatibles in C. Number of internal states in the description of the FSM.

Procedure **inicializa**(T) initializes C to the set of internal states in the initial description T which actually is a closed cover of the FSM. **coste**(C) evaluates the cost of a solution C. There are three main functions in ARNES: **expand**, **irredundant-cover** and **reduction** which we will describe in detail. Once C has been transformed by the application of basic previous functions, procedure **table**(T) builds a symbolic description T' for the FSM defining an internal state for each compatible in C. If cost of new description is smaller than that of initial one the full process is repeated for T'.

expand adds states to each compatible C_i in C, includes those compatibles needed to fulfill closure requirements of C_i^+ (expanded compatible) and eliminates those in C that now are covered. In Figure 2 this procedure is shown using Pidgin-C.

We remark how after we expand a compatible, C is still a closed cover for T. Moreover, expanding is only allowed if it does not mean an increment of C's cardinality. In fact, procedure **expand1** (expanding a compatible) heuristically obtains the expanded compatible that approximately minimizes the final number of compatibles in C.

irredundant-cover verifies if C is redundant. This is, if any proper subset of C is also a closed cover for T.

reduction transforms C in a new C* where each C_i is sequentially substituted by $C_i^- \subseteq C_i$, such that

$$\{C - C_i\} \cup C_i^- \quad \textit{is a closed cover}$$

```
expand(C,T)
/* given a closed cover of T, returns a closed cover
   composed by larger compatibles, eventually
   prime compatibles, without having generated
   them previously*/
{  C = order(C); /* order compatibles according
   to a heuristic rule*/
   Φ* = Φ = coste(C);
   for(i = 1; i ≤ |C|; i++)
   {(W, DIMP, Ci+) = expand1(Ci,);
      C = (C ∪ Ci+ ∪ DIMP) − Ci − W;
      Φ = coste(C);
      /* Ci+ is a compatible which contains Ci */
      /* DIMP is the closure set Ei+ of Ci+ minus
         those compatibles in C */
      /* W is the set of compatibles in C contained
         in Ci+ or in any compatible in DIMP*/
   }
   if(Φ < Φ*)
   {expand(C,T);
   }
}
```

Figure 2: Description Pidgin – C of procedure **expand**

Reduction allows to move away from a solution to another one of less cost as the application of the whole procedure to state table T' corresponding to C* may lead to less cost solutions. Moreover, reduction eliminates states that are covered by more than one compatible. The minimization of such number of states leads to the maximization of don't cares in excitation functions. In Figure 3 we described the procedure.

```
reduction(C,T)
  /* returns C* where each compatible Ci is substituted
  sequentially by Ci−, minimum compatible which
  when substituting Ci, verifies that C* is still a closed
  cover of T */
  {
     C = order2(C);
     for (i = 1; i < |C|; i++)
     {  H = {C − (Ci)} ∩ Ci;
        /* H states in Ci covered by at least one other
        compatible in C */
        if(H != ∅)
        {  C_i = Ci − MCEE(H); /* largest set of states
           that may be eliminated*/
           C = (C − Ci) ∪ C_i;
        }
     }
  }
```

Figure 3: Description Pidgin – C of procedure **reduction**

Several strategies have been developed by us in order to achieve efficient state reduction. Mainly, those concerning the way in which compatibles are expanded aiming at maximizing the difference of the number of compatibles covered and that of compatibles implied by the expanded one. Also ordering strategies are important because many of the procedures described are order dependent.

Algorithm structure reminds that of ESPRESSO-IIC [13] but operations performed by our procedure are different from those there. Note that as well as covering constraints closure ones play an essential role in the state reduction of FSMs.

Results

The new algorithm has been coded in C and a large set of FSMs used to test it. As we have no knowledge of any specific standard benchmark for state reduction we have opted for selecting a set of machines from the literature (all of them referenced in [14]). *Table I* shows the cardinality of the solutions obtained by different heuristic approaches for these machines. G1[15] stands for the first solution of a general branch and bound algorithm applied to solve the 0-1 linear program problem which expresses the covering and closure constraints. The algorithms referenced as BENNETS [11] and REDUCES [12] are both based on MCs. We also include a column for the new algorithm and one for the minimum. In Table I new algorithm obtains minimum cardinality solutions in 14 of 15 cases, including the well known 22 state machine in [21]. Time comparation is no significant because all the used algorithms are very fast. Moreover, when dealing with small machines generation of complete sets of MCs or PCs usually is not computationally expensive. We can conclude for our experience with a large number of machines that the new state reduction algorithm gets high quality solutions and is little time consuming. Experiments on a set of large random generated machines has been performed concluding that the new algorithm is superior to previous reported algorithms both for small and large machines.

Conclusions

Our contribution focuses on the synthesis of PLA-based Finite State Machines. In particular, attention is paid to state reduction which convenience in optimal design of FSMs starting from behavioral descriptions has been suggested. We have developed and programmed an algorithm, ARNES, which heuristically reduces the number of states in symbolic descriptions of FSMs leading to near-minimal FSMs which are them assigned using an optimal state assignment program. It should be considered an intermediate step towards the concurrent treatment of state minimization and state assignment which will solve the optimal FSM realization problem.

FSM	Parameters			G1 PCs	BENNETS MCs	REDUCES MCs	New Algorithm	MINIMUM
	ni	*ns*	*no*	n'_{sG}	n'_{sB}	n'_{sR}	n'_s	
FSM1 [16]	4	5	1	4	4	4	3	3
FSM2 [17]	4	9	1	5	6	5	4	4
FSM3 [18]	4	7	1	4	4	4	4	4
FSM4 [19]	3	6	1	4	3	3	3	3
FSM5 [20]	3	6	1	3	4	4	3	3
FSM6 [21]	4	22	1	70	12	11	9	9
FSM7 [22]	4	9	1	6	5	5	4	4
FSM8 [23]	4	6	1	3	4	4	3	3
FSM9 [24]	6	9	1	4	4	4	3	3
FSM10 [25]	7	8	1	5	5	5	5	4
FSM11 [19]	3	6	1	5	5	5	4	4
FSM12 [11]	4	9	1	5	4	4	3	3
FSM13 [26]	5	6	1	4	3	3	3	3
FSM14 [27]	3	6	1	2	2	2	2	2
FSM15 [28]	5	10	1	5	4	4	4	4

ni: # of symbolic inputs in the FSM;
ns: # of states;
no: # of outputs;
n'_{sG}: *cardinality of the solutions by G1;*
n'_{sB}: *cardinality of the solutions by BENNETS;*
n'_{sR}: *cardinality of the solutions by REDUCES;*
n'_s: *cardinality of the solutions by new algorithm;*

Table I

References

[1] J. Hopcroft, "An *nlogn* Algorithm for Minimizing States in a Finite Automaton", in *Theory of Machines and Computation*, Kohavi ed., pp. 189-196, Academic Press 1971.

[2] C. Pfleeger, "State Reduction of Incompletely Specified Finite State Machines", *IEEE Trans. on Computers*, pp. 1099-1102, Dec. 1973

[3] C. J. Tseng, A. M. Prbhu, C. Li, Z. Memood, and M.M. Tong, "A Versatile Finite State Machine Synthesizer," *Proc. 1986 Int. Conf. on CAD*, pp. 206-209.

[4] B. M. Pangrle and D. Gajski, "Design Tools for Intelligent Silicon Compilation," *IEEE Trans. on Computer-Aided Design*, vol. CAD-6, No. 6, Nov. 1987.

[5] R. Brayton, R. Camposano, G. De Micheli, R. Otten and J. van Eijndhoven, "The Yorktown Silicon Compiler System," in *Silicon Compilation*. D. Gajski (ed.), Addison Wesley, 1988.

[6] M. A. Perkowski, J. Liu, "Generation and Optimization of Finite State Machines from Parallel Program Graphs", *DIADES Research Group Report 10/89*, Dept. EE PSU.

[7] G. De Micheli, "Synthesis of Control Systems", in *Design Systems for VLSI Circuits*, Logic Synthesis and Silicon Compilation, G. De Micheli, A. Sangiovanni-Vincentelli and P. Antognetti eds., pp. 327-364, Martinus Nijhoff Publishers, 1987.

[8] H. Kubo, "A Procedure for Generating Test Sequences to Detect Sequential Circuit Failures", *NEC Res. Dev.* 12: 69-78, 1968.

[9] H.D. Schnurmann, E. Lindbloom and R.G. Carpenter, "The Weighted Random Test-Pattern Generator", *IEEE Trans. on Computers*, July 1975.

[10] S. Devadas, H-K. Tony Ma, A. Richard Newton and A. Sangiovanni-Vincentelli, "Synthesis and Optimization Procedures for Fully and Easily Testable Sequential Machines", *1988 International Test Conference*, pp. 621-630.

[11] R. G. Bennetts, J. L. Washington and D. W. Lewin, "A Computer Algorithm for State Table Reductions," *IERE Radio and Electron. Eng.*, Vol. 42, pp. 513-520, Nov. 1972.

[12] M. J. Avedillo, J. M. Quintana and J. L. Huertas, "A New Method for the State Reduction of Incompletely Specified Sequential Machines", accepted for publication in the *1990 European Design Automation Conference, EDAC'90*, to be held in Glasgow (U.K), 12-15 March 1990.

[13] R. K. Brayton, G.D. Hatchel, C. McMullen, and A.L. Sangiovanni, "Logic Minimization Algorithms for VLSI Synthesis". Hingham, MA, Kluwer Academic Pub., 1984.

[14] B. Reusch and W. Merzenich, "Minimal Coverings for Incompletely Specified Sequential Machines", *Acta Informatica*, No. 22, pp. 663-678, 1986.

[15] B. E. Gillet, "Introduction to Operations Research," McGraw Hill, 1976.

[16] S. H. Unger, "Asynchronous Sequential Switching Circuits", Wiley-Interscience, New York, 1969.

[17] C. V .S. Rao, N.N. Biswass, "Minimization of Incompletely Specified Sequential Machines," *IEEE Trans. on Computers*, vol. C-24, pp. 1089-1100. November 1975.

[18] N. N. Biswas, "State Minimization of Incompletely Specified Sequential Machines," *IEEE Trans. on. Computers*, Vol. C-23, pp. 80-84, Jan. 1974.

[19] A. D. Friedman and R. Menon, "Theory & Design of Switching Circuits," Computer Science Press, Inc., 1975.

[20] A. Grasselli and F. Luccio, "Some Covering Problems in Switching Theory.""Network and Switching Theory", Academic Press, 1968.

[21] S. House, "A New Rule for Reducing CC Tables," *IEEE Trans. on Computers* (Short Notes), November 1970.

[22] A. Grasselli, "Minimal Closed Partitions for Incompletely Specified Flow Tables," *IEEE Trans. on Computers* (Short Notes), pp. 245-249, April 1966.

[23] S. C. De Sarkar, A.K. Basu and A.K. Choudhury, "Simplification of Incompletely Specified Flow Tables with the Help of Prime Closed Sets," *IEEE Trans. on Computers* (Short Notes), Vol. C-18, pp 953-956, October 1969.

[24] H. D. Ehrich, "A Note on State Minimization of a Special Class of Incomplete Sequential Machines," *IEEE Trans. on Computers* (Short Notes), Vol C-21, pp. 500-502, May 1972.

[25] S. C. De Sarkar, A.K. Basu and A.K. Choudhury, "On the Determination of Irredundant Prime Closed Sets," *IEEE Trans. on Computers* (Short Notes), Vol C-20, pp. 933-938, August 1971.

[26] F. Luccio, "Reduction of the Number of Columns in Flow Table Minimization," *IEEE Trans. on Computers*, pp 803-805, 1966.

[27] F. Luccio, "Extending the Definition of Prime Compatibility Classes of States in Incomplete Sequential Machine Reduction," *IEEE Trans. on Computers*, Vol. C-18, pp. 537-540, June 1969.

[28] E. J. McCluskey, "Minimum-State Sequential Circuits for a Restricted Class of Incompletely Specified Flow Tables," *Bell Syst. Tech. J.*, pp.1759-1768, Nov. 1962.