Bharat K. Bhargava was born in U.P., India, on October 14, 1948. He received the B.Sc. (Honors) degree in mathematics from the Punjals University, India, in 1966, B.E. degree in electrical communication engineering from the Indian Institute of Science, Bangalore, India, in 1969, and the M.S. degree in electrical engineering from Purdue University, West Lafayette, Ind., in 1971. He is currently working towards the Ph.D. degree in electrical engineering at Purdue University.

From 1970 to 1973, he was a Teaching and Research Assistant in the School of Electrical Engineering of Purdue University. In June 1973, he joined the Regenstrief Institute in Indianapolis, Ind., as a Systems Analyst, where he is currently involved in the development of application of computer sciences to health care delivery. He is interested in applications of pattern recognition and computer sciences to health care, business, and sciences.

Mr. Bhargava received the National Scholarship from the government of India from 1963 to 1969 and was awarded the Burmah-Shell Scholarship in 1969 to continue graduate work in electrical engineering at I.I.T., Delhi, India.

State Reduction in Incompletely Specified Finite-State Machines

CHARLES P. PFLEEGER

Abstract—The problem of reducing the number of states in an arbitrary incompletely specified deterministic finite-state machine to k states (for a given k) has proven intractible to solution within "reasonable" time; most techniques seem to require exponential time. Two reduction techniques—state assignment to the DON'T CARE entries, and so-called "state splitting"—are investigated. For both of the techniques, the question, "Can I achieve an equivalent k state machine?" is shown to be polynomial complete, with the resulting conjecture that neither is solvable in time bounded by a polynomial function of the size of the machine.

Index Terms—DON'T CARE conditions, finite automaton, incompletely specified finite-state machine, machine minimization, maximum compatibles, polynomial complete, polynomial reducible, state reduction, state splitting, sequential machine.

Introduction

THERE is much interest in a class of decidable problems known as "polynomial complete" problems. None of these problems is known to be solvable in time bounded by a polynomial function of the size of the input. The problems in the class are so related that if any of these problems possess a deterministic solution in polynomial-bounded time, then all in the class do. Because of the size of the class, it is conjectured [1] that no polynomial time-bounded decision procedure exists for any problem in this class.

Given a completely specified deterministic finite-state machine, there is a deterministic technique which operates within polynomial time to determine if there are two or more equivalent states. This well-known technique involves partitioning the states into equivalence classes, and can be performed in time proportional to $m \cdot n^2$ for a machine having n states and m input symbols (see, for example, Kohavi [2]).

However, in an incompletely specified finite-state machine, the problem of specifying transitions so as to partition the states into at most k equivalence classes of states for a given k has no known polynomial time solution. This problem is shown to be polynomial complete. It is therefore presumed that no solution to this problem can be found in polynomial time using a deterministic computing device.

Theorem 1

The following problem (call it $P1$) is polynomial complete. Given an incompletely specified deterministic finite automaton $M = (K, \Sigma, \delta, q_0, F)$[1] and $k > 0$. Is there a way to assign a state to each unspecified transition so that the resulting complete automaton has at most k equivalence classes of states?

Definitions

If P and Q are two problems, then P is *polynomial reducible* to Q iff the following condition holds: there is a transformation f which can be performed in deterministic polynomial-bounded time such that x is a solution of P iff $f(x)$ is a solution of Q. The implication here is that Q is not harder (within a polynomial difference) than P to solve.

A problem P is *polynomial complete* iff 1) it can be solved by a nondeterministic Turing machine which operates in time

Manuscript received January 22, 1973; revised May 11, 1973.
The author is with the Department of Computer Science, Pennsylvania State University, University Park, Pa. 16802.

[1] K and Σ are finite sets of "states" and "inputs," respectively; δ, called the "transition function," is a mapping from a subset of $K \times \Sigma$ into K; the "initial state" q is in K; and the set of "final states" F is a subset of K.

bounded by a polynomial function of the length of its input tape, and 2) all other problems solvable nondeterministically in polynomial time are polynomial reducible to P.[2]

The proof of this theorem centers on the reduction of the following problem to the state assignment problem $P1$. Let $G = (V, E)$[3] be a graph with vertices $V = \{v_1, \cdots, v_m\}$. We say a function Φ *colors* G with c colors iff $\Phi: v \rightarrow \{1, \cdots, c\}$, and $\Phi(v) \neq \Phi(v')$ if $\{v, v'\} \in E$. Karp [3] has shown that the following problem is polynomial complete. Given G and c, does such a coloring function Φ exist? Call this problem the "graph coloring problem."

Construction

Let G and c be given. Assume without loss of generality that G has no loops nor isolated vertices.[4] Construct the finite automaton $M = (K, \Sigma, \delta, S_0, \{S_F\})$ where

$$K = V \cup \{S_0, S_N, S_F\} \qquad \text{(and none of } S_0, S_N, S_F \in V)$$

$$\Sigma = \{a_i \mid v_i \in V\}.$$

δ is defined as follows:

$$\begin{aligned}
\delta(S_0, a_i) &= v_i \quad \forall a_i\\
\delta(S_N, a_i) &= S_N \quad \forall a_i\\
\delta(S_F, a_i) &= S_F \quad \forall a_i\\
\delta(v_i, a_i) &= S_F \quad \forall a_i\\
\delta(v_i, a_j) &= S_N \text{ iff } \{v_i, v_j\} \in E.
\end{aligned}$$

See Fig. 1 for an example of the construction of M from a given graph. Notice, for example, that vertex v_1 is connected to vertex v_2, and therefore $\delta(v_1, a_2) = S_N$. $\delta(v_1, a_3)$ is undefined. Obviously M is incompletely specified iff G is not the complete graph[5] on vertices V.

Lemma 1

If G is colorable with c colors, then δ may be extended so that the resulting complete automaton has at most $c + 3$ equivalence classes of states.

Proof: Let Φ color G with c colors. Construct $M' = (K, \Sigma, \delta', S_0, \{S_F\})$ where δ' is defined as follows. For each $v, v_j \in V$,

$$\delta'(S_0, a_j),\quad \delta'(S_N, a_j),\quad \delta'(S_F, a_j) \text{ are defined as for } \delta.$$

$$\delta'(v, a_j) = S_N \text{ iff for some } v'\text{: } \{v', v_j\} \in E \text{ and } \Phi(v') = \Phi(v);$$

$$\delta'(v, a_j) = S_F \text{ otherwise.}$$

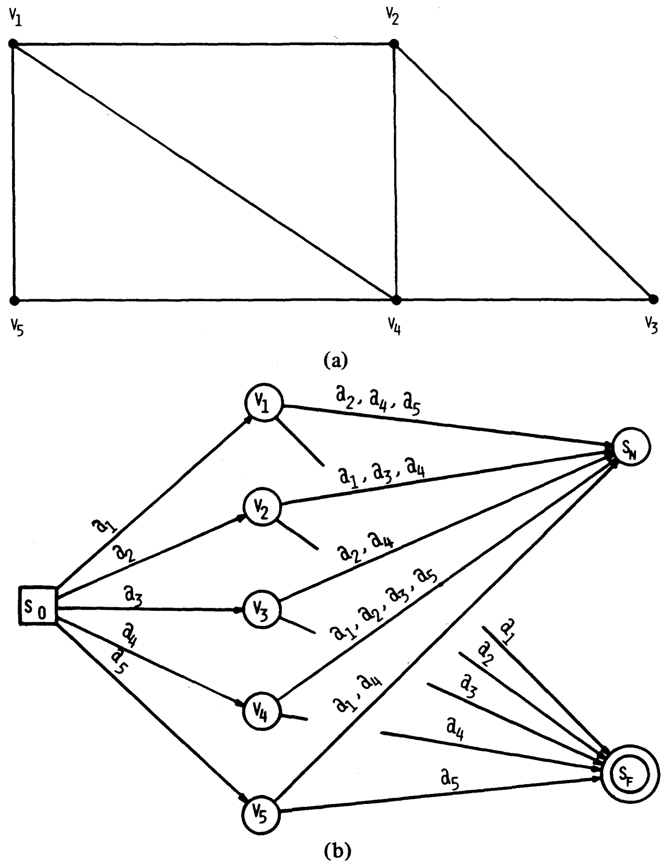


Fig. 1. Example of construction of M. (a) Given a graph. (b) Construction of machine M from this graph.

First it is shown that this function is an extension of δ. To see this fact, consider any i and j for which $\delta(v_i, a_j)$ is defined.

Case 1: $\delta(v_i, a_j) = S_N$. Then $\{v_i, v_j\} \in E$. Letting $v' = v$, $\delta'(v_i, a_j) = S_N$.

Case 2: $\delta(v_i, a_j) = S_F$. Then $i = j$. Suppose $\delta'(v_i, a_j) = S_N$, which would imply that there is a v': $\{v', v_j\} \in E$ and $\Phi(v_i) = \Phi(v')$. But since $i = j$, this last equality becomes $\Phi(v_j) = \Phi(v')$. But $\{v', v_j\} \in E$, which contradicts the fact that Φ colors G. Hence, $\delta'(v_i, a_j) = S_F$.

Thus, for any i and j, if $\delta(v_i, a_j)$ is defined, then $\delta'(v_i, a_j) = \delta(v_i, a_j)$, and so δ' is an extension of δ.

Next it will be shown that for any i and j, if $\Phi(v_i) = \Phi(v_j)$, then $\delta'(v_i, a_k) = \delta'(v_j, a_k)$ for all k, and therefore states v_i and v_j are equivalent. Let $\Phi(v_i) = \Phi(v_j)$ and assume there is some k for which $\delta'(v_i, a_k) \neq \delta'(v_j, a_k)$. Without loss of generality, assume $\delta'(v_i, a_k) = S_F$ and $\delta'(v_j, a_k) = S_N$.

$$\delta'(v_i, a_k) = S_F \text{ implies } \forall\, v',\ \{v', v_k\} \notin E \text{ or } \Phi(v_i) \neq \Phi(v').$$

$$\delta'(v_j, a_k) = S_N \text{ implies } \exists\, v',\ \{v', v_k\} \in E \text{ and } \Phi(v_j) = \Phi(v').$$

Now by assumption $\Phi(v_i) = \Phi(v_j)$, and so $\Phi(v_i) = \Phi(v')$. $\{v', v_k\} \in E$ and $\Phi(v_i) = \Phi(v')$ contradicts the fact that $\delta'(v_i, a_k) = S_F$.

By this contradiction, for any i and j, if $\Phi(v_i) = \Phi(v_j)$, then

[2]This definition does not lead to a circularity problem. The original definition of polynomial complete in [1] posed a problem solvable nondeterministically in polynomial time; it was then shown that if this problem could be solved deterministically in polynomial time, then the classes of nondeterministic and deterministic polynomial time-bounded computations would be equal.

[3]V is a finite set of "vertices," and E is a subset of $V \times V$, called the "edges" of G.

[4]A loop is an edge which connects a vertex to itself. An isolated vertex is one which is not connected to any other vertices.

[5]The complete graph is one in which each vertex is connected to each other vertex.

$\delta'(v_i, a_k) = \delta'(v_j, a_k)$ for all k, and thus states v_i and v_j are equivalent.

The states of M' which are vertices in G are partitioned into at most c equivalence classes by Φ. M' has three additional states: S_0, S_N, S_F, so M' has at most $c + 3$ equivalence classes of states. Therefore M' is the required extension of M. □

Lemma 2

If M may be completed so that the resulting reduced automaton has k states, then G may be colored with $k - 3$ colors.

Proof: It is easily seen from the definition of δ that none of S_0, S_N, and S_F are compatible, and that no $v \in V$ is compatible with any of S_0, S_N, and S_F. Let M'' be the reduced automaton obtained after specifying all of δ in M. The states of M which are vertices in G are partitioned into $k - 3$ equivalence classes in M'', so let the states of M'' be $\{S_0\}$, $\{S_N\}$, $\{S_F\}$, and $C_1, \cdots, C_{k-3}$. For each $v \in V$, define

$$\Phi(v) = i \qquad \text{iff } v \in C_i.$$

The lemma is proved by showing that Φ colors G with $k - 3$ colors.

Assume Φ is not a proper coloring function, i.e., assume there are i and j such that $\Phi(v_i) = \Phi(v_j)$ and $\{v_i, v_j\} \in E$. Then for some p, v_i, $v_j \in C_p$. From the definition of M, $\delta(v_i, a_j) = S_N$, since $\{v_i, v_j\} \in E$. However, $\delta(v_j, a_j) = S_F$. Thus v_i and v_j are not compatible, so $v_i \notin C_p$ or $v_j \notin C_p$. By contradiction, Φ is a proper coloring function for G. Since there are exactly $k - 3$ of the C_i, then Φ colors G with $k - 3$ colors. □

Proof of Theorem 1: $P1$ can be solved nondeterministically in polynomial time as follows. First, nondeterministically generate a transition for each unspecified transition in M. In polynomial time, one may deterministically construct the reduced machine equivalent to this complete one. Then determine if the state set of this reduced machine is of size greater than k. If not, accept M. Thus, this problem can be solved in polynomial time by a nondeterministic Turing machine.

By Lemmas 1 and 2, the graph-coloring problem is polynomial reducible to $P1$. Therefore, $P1$ is polynomial complete. □

The extension from an incompletely specified finite automaton to an incompletely specified sequential machine is natural. Machine M may be converted to a sequential machine by defining an output function which is zero for final states, and nonzero for nonfinal states. By a construction similar to that used in the theorem, the problem of state specification for incompletely specified sequential machines is also polynomial complete.

The result which has just been shown is that the problem of deciding if we can reduce an incompletely specified finite-state machine to a given size by merely specifying a transition for each unspecified transition and reducing the resulting machine is polynomial complete. There is an algorithm, explained in Kohavi [1] (due primarily to Paull and Unger [4]) which can be used to reduce arbitrary incompletely specified finite-state machines. The technique involves "splitting" the properties of one state in the original machine across two or more states in the resultant machine. It is now shown that deciding if one can obtain a machine of size k for a given k, by use of the state-splitting technique, is also polynomial complete.

Theorem 2

The following problem (call it $P2$) is polynomial complete. Given an incompletely specified deterministic finite automaton $M = (K, \Sigma, \delta, S_0, F)$ and $k > 0$. By applying the state-splitting technique, can an automaton $M' = (K', \Sigma, \delta', S'_0, F')$ be obtained, where K' has at most k states, and the set accepted by M' is identical with that accepted by M?

Definitions

A set $C \subseteq K$, $C = \{v_a, \cdots, v_b\}$ is a *compatible* iff $v_i, v_j \in C$ implies [$\forall\, x \in \Sigma, \delta(v_i, x) \in F \Longleftrightarrow \delta(v_j, x) \in F$, whenever both are defined.]

A pair (v_i, v_j) is a *compatible pair* iff $\{v_i, v_j\}$ is a compatible.

A compatible C_i is *maximal* iff $\forall\, v \notin C_i$, $C_i \cup \{v\}$ is not a compatible.

A set C of compatibles $C = \{C_r, \cdots, C_q\}$ is *closed* iff $\forall\, C_i \in C$, $\forall\, v, v' \in C_i$, $\forall\, x \in \Sigma$, $\exists\, C_j \in C$: $\delta(v, x) \in C_j$, and $\delta(v', x) \in C_j$ (i.e., members of C, the set of compatibles, map back into members of C under the operation δ).

The state-splitting algorithm requires selection of a closed set of maximal compatibles which covers all internal states, i.e., each state in the original machine is contained in at least one of the compatibles. These compatibles represent states of the resultant machine, so an equivalent machine of k states may be constructed iff there is a selection of k such maximal compatibles which is closed and which covers all states.

The same construction will be used. That is, given $G = (V, E)$ a graph, where $V = \{v_1, \cdots, v_n\}$. Construct $M = (K, \Sigma, \delta, S_0, \{S_F\})$ where

$$K = V \cup \{S_0, S_N, S_F\} \quad (\text{and none of } S_0, S_N, S_F \in V)$$

$$\Sigma = \{a_i \mid v_i \in V\}$$

$$\delta(S_0, a_i) = v_i, \quad \delta(S_N, a_i) = S_N, \quad \delta(S_F, a_i) = S_F$$

$$\delta(v_i, a_i) = S_F, \quad \delta(v_i, a_j) = S_N \text{ if } \{v_i, v_j\} \in E.$$

Lemma 3

If G is c colorable, then the state-splitting algorithm can produce a machine having at most $c + 3$ states.

Proof: Obviously S_0, S_N, and S_F are incompatible with all other states. Let Φ color G. Define

$$\Phi_i = \{v \in V \mid \Phi(v) = i\}, \qquad \text{for } i = 1, 2, \cdots, c.$$

For each Φ_i, construct K_i as follows.

1) K_i contains Φ_i.

2) For each compatible pair (v, w), where $v \in K_i$ and $w \notin K_i$, add w to K_i iff $\forall w' \in K_i, (w, w')$ is also a compatible pair.

3) Repeat step 2) until no new elements can be added to K_i.

Now consider the time required to execute this algorithm.

If M has n states, there are at most n^2 compatible pairs, and each K_i has at most n members. Thus, time to construct any given K_i requires time bounded by n^3. There are at most n of the K_i, and thus this algorithm requires time at worst proportional to n^4.

Let $M' = (K', \Sigma, \delta', \{S_0\}, \{\{S_F\}\})$ where

$$K' = \{\{S_0\}, \{S_N\}, \{S_F\}, K_1, \cdots, K_c\}$$

$$\delta'(\{S_N\}, a_i) = \{S_N\}, \qquad \delta'(\{S_F\}, a_i) = \{S_F\}$$

$$\delta'(\{S_0\}, a_i) = K_j \text{ iff } [v_i \in K_j, \text{ and } \forall j' < j, v_i \notin K_{j'}].$$

$$\delta'(K_j, a_i) = \{s \in K \mid \text{there exists } v \in K_j\text{: } \delta(v, a_i) = s\}.$$

(Note that s in this set is chosen from $\{S_N, S_F\}$.)

We claim that M' can be produced by the state-splitting algorithm. The following four facts prove this claim.

1) Each of the K_i is a compatible.

Assume otherwise. Then $\exists\ v_p, v_m \in K_i$, and $a_j \in \Sigma$: $\delta(v_p, a_j) = S_F$, and $\delta(v_m, a_j) = S_N$. $\{v_p, v_m\} \subseteq K_i$ iff $\{v_p, v_m\} \subseteq \Phi_i$, or at least one of $\{v_p, v_m\}$ was added to K_i in step 2) of the construction of K_i.

Case 1: $\{v_p, v_m\} \subseteq \Phi_i$. $\delta(v_p, a_j) = S_F$ implies $p = j$. $\delta(v_m, a_j) = S_N$ implies $\{v_m, v_j\} \in E$. But $\Phi(v_m) = \Phi(v_{j=p})$ contradicts the assumption that Φ colors G. Thus $\{v_p, v_m\} \not\subseteq \Phi_i$.

Case 2: Assume v_p was added to K_i after v_m. But (v_p, v_m) is not a compatible pair, and thus v_p could not have been added after v_m.

By this contradiction, each of the K_i must be a compatible.

2) Each of the K_i is maximal.

By the construction, each of the K_i was expanded until it could not be further expanded.

3) K' covers K (i.e., each state in K is contained in at least one of the state sets of K').

This is obvious for S_0, S_N, S_F. Each $v \in V$ is contained in one of the Φ_i since Φ colors G. Since each K_i contains Φ_i, then $v \in K_i$.

4) K' is closed under δ.

It must be shown that $\forall\ x \in \Sigma, \forall\ s, t \in K_i, \exists j\text{: } \delta(s, x) \in K_j$, and $\delta(t, x) \in K_j$. By definition of δ, $\forall\ v \in V$, $\delta(v, x)$ is either S_N or S_F. Assume there is some K_i, some $s, t \in K_i$, and some $x \in \Sigma$ such that $\delta(s, x) = S_N$ and $\delta(t, x) = S_F$. But this was disproven by fact 1) above. Therefore, K' is closed under δ.

$M' = (K', \Sigma, \delta', \{S_0\}, \{\{S_F\}\})$ is the required automaton, thus proving the lemma. □

Lemma 4

If the state-splitting algorithm produces a machine of k states, then G may be colored with $k - 3$ colors.

Proof: Without loss of generality, let

$$M'' = (\{K_1, \cdots, K_{k-3}, \{S_0\}, \{S_F\}, \{S_N\}\}, \Sigma, \delta'', \{S_0\}, \{\{S_F\}\})$$

be the machine of k states which was produced by the state splitting algorithm. Define

$$\Phi(v) = i \text{ iff } v \in K_i, \text{ and } \forall\ j < i, v \notin K_j.$$

Claim: Φ colors G with $k - 3$ colors. Assume otherwise. Then for some v_i, v_j, $\Phi(v_i) = \Phi(v_j)$, and $\{v_i, v_j\} \in E$. Then there is some p such that $v_i, v_j \in K_p$. But $\delta(v_i, a_j) = S_N$ since $\{v_i, v_j\} \in E$. Also, $\delta(v_j, a_j) = S_F$. Then K_p is not a compatible, and M'' could not have been the machine produced by the state-splitting algorithm. By contradiction, Φ colors G with $k - 3$ colors, since there are $k - 3$ of the K_i, and each $v \in V$ is contained in at least one of the K_i. □

Proof of Theorem 2: A nondeterministic machine will generate a set of at most k subsets of K. In deterministic polynomial time, we can test each of these subsets to determine if they are indeed compatibles, that they cover all of K, and that they are closed under the function δ. If yes, then we have produced the desired machine. Thus, the machine derived through state splitting can be produced nondeterministically within time bounded by a polynomial function of the size of the original machine.

By Lemmas 3 and 4, the graph coloring problem is polynomial reducible to $P2$. Therefore, $P2$ is polynomial complete. □

The result of this paper is that the problem of reducing an arbitrary incompletely specified finite-state machine to a given size k is of extreme difficulty (probably exponential) when attempted by two seemingly efficient algorithms. The fact that both of these techniques are polynomial complete suggests that a search for good heuristics and good algorithms for restricted classes of incompletely specified finite-state machines is in order.

Acknowledgment

The author thanks Dr. N. D. Jones for his useful advice on this paper.

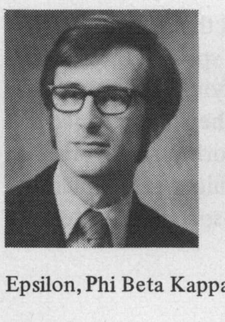
References

[1] S. Cook, "The complexity of theorem-proving procedures," in *Conf. Rec. 3rd Ass. Comput. Mach. Symp. Theory of Computing*, 1970, pp. 151–158.

[2] Z. Kohavi, *Switching and Finite Automata Theory*. New York: McGraw-Hill, 1970, pp. 287–288.

[3] R. Karp, "Reducibility among combinatorial problems," in *Complexity of Computer Computations*, R. E. Miller and J. W. Thatcher, Ed. New York: Plenum, 1972, pp. 85–103.

[4] M. Paull and S. Unger, "Minimizing the number of states in incompletely specified sequential switching functions," *IRE Trans. Electron. Comput.*, vol. EC-8, pp. 356–366, Sept. 1959.



Charles P. Pfleeger was born in Racine, Wis., on March 23, 1948. He received the B.A. degree with honors in mathematics from Ohio Wesleyan University, Delaware.

Since 1970 he has been associated with the Department of Computer Science, Pennsylvania State University, University Park, as a Graduate Teaching Assistant and as an Instructor. He is also currently working toward the Ph.D. degree in computer science.

Mr. Pfleeger is a member of Upsilon Pi Epsilon, Phi Beta Kappa, and the Association for Computing Machinery.