Comments on “Closure Partition Method for Minimizing Incomplete Sequential Machines”

C.-C. Yang and Marek K. Babiniski

Abstract—This correspondence provides some clarifications and revision to the above-mentioned paper. It also presents the generalized versions of the definitions and theorems related to closure classes. These versions are applicable to arbitrary subsets of the state set of a sequential machine.

Index Terms—Closure class, implied block, sequential machine.

INTRODUCTION

As seen from the above-mentioned paper and [1], [2], the concept of closure class is not only useful for minimizing an incomplete sequential machine, but also in solving other problems [1], [2]. However, a closure class originally defined for a compatible needs to be extended to an arbitrary subset of the state set of a sequential machine. As this extension is made, the properties of the generalized closure classes which have not been fully developed in previous work would require more considerable attention.

This correspondence provides clarifications and revision to some typographical errors and unclear points in the paper. The definitions and theorems related to closure classes are generalized. The notations used here follow those of the paper.

CLARIFICATIONS AND REVISION

Clarification 1: The inclusion relation between two compatible defined by Definition 3 is exactly the set-theoretic meaning of inclusion of one set in another. Therefore, we should adopt the usual set inclusion symbols \( \subseteq \) and \( \subset \) for defining a subset and a proper subset, respectively. If these appropriate conventions are adopted, Definition 3 could be even omitted. From now on, we will use \( \subseteq \) and \( \subset \) to replace \( \subseteq \) and \( < \), respectively, when the inclusion relation between two compatible blocks (to be extended later) is involved.

Clarification 2: The inclusion relations \( \supseteq \) and \( \supset \) between two closure classes \( E_i \) and \( E_j \) do not follow the exact meanings of the set inclusion relations \( \supseteq \) and \( \supset \) unless \( E_i \) and \( E_j \) are closure independent classes. Thus, we still use the symbols \( \supseteq \) and \( \supset \). The equality case \( E_i^{'}, E_j^{'}, E_i \supset E_j \) with respect to two distinct compatible \( E_i \) and \( E_j \), without the set inclusion property is separately considered in Definition 7 for establishing an equivalence relation so that Definition 4 in fact defines only the inclusion relation \( E_i \supset E_j \) where \( E_i \) and \( E_j \) are respectively, with respect to, \( C_i \) and \( C_j \) such that \( C_i \supset C_j \) but not with respect to arbitrary \( C_i \) and \( C_j \).

Unfortunately, this important restriction is not explicitly included in the definition. With this restriction for \( E_i \supset E_j \), Theorem 2 is correct (as will be reverified later). Otherwise, the second implication in Theorem 2 is wrong as evidenced by the counter example below:

\[
D_{TTT} \supset D_{GG} \quad \text{but} \quad 1,2,3,8 \supset 4,5. 
\]

Thus, if Definition 4 is retained, then Theorem 2 should be revised as \( C_i \supset C_j \) implies \( E_i \supset E_j \). Otherwise, we should revise Definition 4 as follows.

Revised and extended form of Definition 4: A closure class \( E_i \) with respect to \( B_i \) contains a distinct closure class \( E_j \) with respect to \( B_j \), written \( E_i \supset E_j \), if and only if \( B_i \supset B_j \) and every element of \( E_j \) is contained in some element of \( E_i \) (where \( B_i \) and \( B_j \) are subsets of the state set \( Q \) and called blocks).

Clarification 3: Since every (unity) set containing a single present or next state is not required for a closure test when the closure condition of a cover is examined, the closure condition for a cover can be alternatively defined as follows.

1) A cover is closed if and only if for each element in the cover, the set of all next states for every input is contained in some element of the cover.

2) A cover is closed if and only if for each element containing at least two states in the cover, the implied compatible (or the implied block to be extended later) for every input is contained in some element of the cover.

However, as the word “cover” in both previous definitions is replaced by the phrase “an arbitrary set of compatibles (or of blocks in an extended version),” the modified definitions are generally not equivalent since such an arbitrary set being closed in the former sense is stronger than that in the latter sense; in other words, the set being closed in the former sense implies the set being closed in the latter sense but the reverse may not be true. As shown in [1], in order to differentiate both of these closure conditions for an arbitrary set, the latter case based on the weaker sense, is referred to as “quasi closed” and the former still as “closed.” In the paper, the closure condition of each closure class, closure related class, or closure aggregate follows the weaker definition for its closure.

REVERIFICATION AND GENERALIZATION

Generalized version [1] of Definition 1: For any block \( B \), the set

\[
B' = M(B(w)) = \bigcup_{q \in B} \{ M(q,w) \} 
\]

with

\[
K(B') \geq 2
\]

for an input word \( w \) in \( S^* \) is called the implied block with respect to \( B \) and for the input word \( w \). If the block \( B' \) is a compatible, then it is called the implied compatible with respect to \( B \) and for \( w \).

By this version, there is no need to differentiate between an implied block obtained directly by some input and another one obtained by repeated use of the transitivity rule. The implied block \( B = M(B,A) \) with respect to \( B \) for and the null input word \( A \) can be used to simplify the subsequent definition for a closure independent block and class. Since (1b) is essential, \( B \) must belong to the set \( 2^5 - Q^* - \{|e| \} \) where \( Q^* \) is the set containing all unity blocks, i.e.,

\[
Q^* = \{|q| q \in Q \}.
\]

Generalized version of Definition 2: A block \( B_i \) is called closure independent if one of the following conditions is satisfied.

Condition 1: Each of the implied blocks with respect to \( B_i \) and for some input word in \( S^* \) is a subset of \( B_i \).

Condition 2: There exists no implied block with respect to \( B_i \) and for all input words in \( S^* \).

Otherwise, \( B_i \) is called closure dependent. The closure class \( E_i \) with respect to a block \( B_i \) is the union of \( \{B_j\} \) and the set containing the implied blocks with respect to \( B \) and for all input words in \( S^* \).
under the max operation, i.e.,
\[ E_i = \max \left\{ |B_i| \cup |M(B_i,w)| \mid w \in S^* \text{ and } K[M(B_i,w)] \geq 2 \right\}. \] (3a)

A closure class with respect to a closure independent block \( B_i \) is called a closure independent class denoted by \( I_i \), and a closure class with respect to a closure dependent block \( B_i \) is called a closure dependent class denoted by \( D_i \).

As consequences of this definition, we have the following Theorems.

**Theorem 1:** If one of the following sets of equations
\[
M(B_i,w) \subseteq B_i \quad \text{for all } w \in S^* \quad (4a)
\]
\[
K(B_i) = 1 \quad (4b)
\]
\[
K(B_i) \geq 2 \quad \text{and } K[M(B_i,w)] \leq 1 \quad \text{for all } w \in S^* - \{\lambda\} \quad (4c)
\]
is satisfied, then \( E_i \) turns out to be \( I_i \) and
\[
I_i = \{B_i\}. \quad (5)
\]

If
\[
M(B_i,v) \not\subseteq B_i \quad \text{and } K[M(B_i,v)] \geq 2 \quad (6)
\]
then \( E_i \) turns out to be \( D_i \) and
\[
D_i = \max \left\{ |M(B_i,w)| \mid \text{each } w \in S^*, M(B_i,v) \supseteq B_i, K[M(B_i,v)] \geq 2 \right\} \quad (7)
\]

where both sides of (8) stand for implied blocks. The equality relation arises whenever \( B_i \) contains \( K(B_i) = K(B_i) \) states \( q^s \)'s whose corresponding final states resulted from a sequence of state-transitions by some input word \( w \) are undefined and/or duplicated with some elements of \( M(B_i,w) \) and \( B_i \) does not contain these states \( q^s \)'s. The inclusion relations for all \( w \) in (8) imply \( D_i \geq D_i \) as seen from (7).

The second implication of Theorem 2 is obvious since first \( E_i \) and \( E_j \) are with respect to \( B_i \) and \( B_j \) where \( B_i \) and \( B_j \) are constrained by the condition of \( B_i \supseteq B_j \) and second \( E_i \) and \( E_j \) must respectively contain \( B_i \) and \( B_j \).

Q.E.D.

On the other hand, if \( B_i \) and \( B_j \) are arbitrary blocks, the second implication in Theorem 2 is incorrect as already shown by the counterexample.

**Generalized version of Theorem 3:** Given a block \( B \) with \( K(B) > 2 \).

If \( B_i \) for all \( i \) are possible subsets of \( B \) with \( K(B) - 1 \) elements, then the closure class \( E_B \) with respect to \( B \) can be determined by
\[
E_B = \max \left\{ \bigcup_{i=1}^{s} E_i \cup |B_i| \right\} \quad (3b)
\]

where \( E_i \) is the closure class with respect to \( B_i \).

**Proof:** Since \( K(B_i) \geq 2 \) and \( M(B_i,\lambda) = B_i \), the form of (2a) can be simplified for \( E_i \) as
\[
E_i = \max \left\{ |M(B_i,w)| \mid \text{each } w \in S^* \text{ and } K[M(B_i,w)] \geq 2 \right\}. \quad (3c)
\]

Similarly,
\[
E_B = \max \left\{ |M(B_i,w)| \mid \text{each } w \in S^* \text{ and } K[M(B_i,w)] \geq 2 \right\}. \quad (3d)
\]

We need only to consider two distinct types of the implied blocks in each closure class, i.e.,

Type 1) \( M(B_i,w) \subseteq B_i \) for some \( u \in S^* \) and

Type 2) \( M(B_i,v) \not\subseteq B_i \) for some \( v \in S^* - \{\lambda\} \) where \( K[M(B_i,w)] \geq 2 \) for \( w = u \) or \( w = v \).

Each first type implied block is deleted from \( E_i \) by the max operation. But each second type implied block must be another element of \( E_i \). If the implied blocks for all input words are of the first type, then (5) is yielded. On the other hand, if there exists at least one implied block of the second type, then (7) is yielded. In the latter case,
\[
D_i \not\supseteq \{B_i\}. \quad (9a)
\]
or equivalently
\[
D_i \supset \{B_i\}. \quad (9b)
\]

Similarly, \( E_B \) follows the similar arguments as just described. Suppose that \( M(B_i,v) \) with respect to \( B \) and for some \( v \in S^* - \{\lambda\} \) is such an element in \( D_B \), then
\[
M(B_i,v) \not\subseteq B \quad (10a)
\]
and
\[
K(B) \geq K[M(B_i,v)] > 2 \quad (10b)
\]
or
\[
K(B) > K[M(B_i,v)] \geq 2 \quad (10c)
\]
where (10b) or (10c) must hold since \( K(B) > 2 \) and \( M(B_i,v) \) is an implied block. Consider the following probable cases.

Case 1: If \( K(B) > K[M(B_i,v)] \geq 2 \), then \( B \) contains at least one state and at most \( K(B) - 3 \) states whose corresponding final states resulted from a sequence of state-transitions by the input word \( v \) are undefined and/or duplicated with some elements in \( M(B,v) \).

Case 2: If \( K(B) = K[M(B_i,v)] \geq 2 \), then \( B \) contains at most \( K(B) - 3 \) states which have the similar resulted final states as described in Case 1.

Case 3: If \( K(B) > K[M(B_i,v)] = 2 \), then \( B \) contains at least one state and at most \( K(B) - 2 \) states which have the similar resulted final states as described in Case 1.

Under any case, if we delete one of those states in \( B \) whose corresponding final state is undefined or duplicated to obtain a proper subset \( B_i \) of \( B \), then
\[
M(B_i,v) = M(B_v) \quad (11)
\]
for the same input word \( v \) as specified in (10). Since \( B_i \) is a proper subset of \( B \) and (10a) and (11) hold, \( M(B_i,v) \) must not be contained in \( B_i \), i.e.,
\[
M(B_i,v) \not\subseteq B_i. \quad (12)
\]

This means that there exists at least one \( i \) such that \( M(B_i,v) \) must be a distinct element in \( E_i \) with respect to \( B_i \). On the other hand, if we delete one state in \( B \) whose final state caused by \( v \) is defined and not duplicated to obtain a proper subset \( B_i' \) of \( B_i \), then
\[
M(B_i',v) \subseteq M(B_i,v). \quad (13)
\]

When the max operation is performed on the union of all \( E_i \) and \( |B| \), the implied blocks \( M(B_i,w) \subseteq B \) of the first type and \( M(B_i,v) \) satisfying (13) are deleted and \( M(B_i,v) \) satisfying (11) is merged with \( M(B,v) \) so that (3b) is established.

Q.E.D.

By means of this theorem, Theorem 3 in the paper simply becomes a Corollary. This theorem also provides an effective technique for finding the closure classes with respect to some larger blocks.
Comments on "Derivation of Minimal Complete Sets of Test-Input Sequences Using Boolean Differences"

GERNOT METZE, DONALD R. SCHERTZ, KILIN TO, GORDON WHITNEY, C. R. KIME, AND JEFFREY D. RUSSELL

In the above paper,[1] the partial Boolean difference is defined and an algorithm for deriving complete minimal test input sequences for single-fault detection is given. In this correspondence, it is shown that the test case for the algorithm is inadequate with respect to the need for multiple path sensitization. It is further shown that the computational procedure used in footnote 1 does not guarantee minimality of the test set and does not permit the extension to fault location as claimed in the paper.

It is well known that the derivation of fault-detection test sets by procedures based on path sensitization may require the consideration of all possible multiple paths from a fault site to an observable output [1]. Yet, the "partial Boolean differences algorithm" in footnote 1 appears to be able to derive complete test sets by computing partial Boolean differences along single paths only.

The development of the partial Boolean differences algorithm begins with the definition of the partial Boolean difference and the introduction of the concept of "functional path" on p. 26 and 27 of footnote 1. The functional path defined by (17) and illustrated in Fig. 1 is clearly a single path. A possible extension of (17) to partial Boolean differences with respect to more than one variable is mentioned, but except for the statement of (20) reference is made to when and why multiple paths from fan-out points at internal nodes or primary inputs might have to be sensitized.

In fact, the discussion pertaining to the network of Test Fig. 2 on p. 29 of footnote 1 specifically states that "a careful inspection of the switching network will reveal the existence of ten paths..." and consideration of these ten paths is then detailed. These ten paths are clearly the single paths of the network; since the network exhibits reconvergent fan-out from an internal node as well as from the primary inputs, it also contains a large number of multiple paths, but these multiple paths are not considered. Thus one might conclude that the computation of partial Boolean differences along single paths suffices for all such networks. This conclusion is strengthened just prior to the formal statement of the partial Boolean differences algorithm on p. 31 of footnote 1, when the claim is made with reference to Roth et al. [1] that "by making use of the partial Boolean difference concept, the network topology enters directly in the design of the desired test-input sequence, and the question of single versus multiple path sensitization is automatically satisfied [1]." However, consider for example the network of Fig. 3 in the Roth et al. paper [1] referred to above. The partial Boolean differences algorithm fails to find a test for the fault "output of gate 6 stuck-at-1" and therefore identifies it as "inconsequential." But input vectors 0110, 0111, or 1110 detect this fault. Thus, the partial Boolean differences algorithm is not correct since it will fail to compute tests for those faults whose detection requires multiple path sensitization.

Yet another problem arises from the attempt to use the fault table information for fault location purposes. In Table III of the paper, Marinos claims that the faults "line 6 s-a-1," "line 7 s-a-1," and "line 8 s-a-1" are distinguishable. Indeed, since these lines are the input and output lines of an or gate, the s-a-1 faults on them can never be distinguished. This problem, of course, also results from the incompleteness of the fault table information derived.

It should be noted that the converse problem also arises when the partial Boolean differences algorithm is used for the identification of faults detected by a selected test. Yau and Tang [2] have shown, in a paper published since the submission of our Comments, that all faults on a particular path used to generate a given test pair are not necessarily detected by the test pair. Thus, a fault detected by means of the partial Boolean difference concept outlined in the paper may not only fail to list faults that are actually detected, but also list faults that are in fact not detected.

In summary, this paper as it stands gives a misleading impression of the usefulness of the partial Boolean difference for fault test derivation. The exclusive use of examples in which single path sensitization suffices and the wording of the algorithm suggest that the procedure outlined in the paper is superior to other fault test derivation techniques. In fact, the procedure as it stands does not work for networks having faults whose detection requires multiple path sensitization.

If the procedure is modified to include a consideration of multiple paths, as it must be, it will certainly encounter the same (or larger) computational complexities as any other procedure based on the path sensitization concept [3]. In addition, the fault table information provided by the partial Boolean differences algorithm presented is incomplete and the claims that the procedure derives a minimal fault detection test set and can be extended to derive fault location test sets are incorrect.

REFERENCES