THE ALGORITHMIC COMPLEXITY OF MODULAR DECOMPOSITION

JAN C. BIOCH

ERIM REPORT SERIES *RESEARCH IN MANAGEMENT*	
ERIM Report Series reference number	ERS-2001-30-LIS
Publication	May 2001
Number of pages	15
Email address author	Bioch@few.eur.nl
URL (electronic version)	http://www.eur.nl/WebDOC/doc/erim/erimrs20010614160249.pdf
Address	Erasmus Research Institute of Management (ERIM) Rotterdam School of Management / Faculteit Bedrijfskunde Erasmus Universiteit Rotterdam P.O. Box 1738 3000 DR Rotterdam, The Netherlands Phone: +31 10 408 1182 Fax: +31 10 408 9640 Email: info@erim.eur.nl Internet: www.erim.eur.nl

BIBLIOGRAPHIC DATA AND CLASSIFICATIONS		
Abstract	Modular decomposition is a thoroughly investigated topic in many areas such as switching theory, reliability theory, game theory and graph theory. We propose an *O(mn)*-algorithm for the recognition of a modular set of a monotone Boolean function *f* with m prime implicants and r variables. Using this result we show that the computation of the modular closure of a set can be done in time *O(mn2).* On the other hand, we prove that the recognition problem for general Boolean functions is NP-complete. Moreover, we introduce the so called generalized Shannon decomposition of a Boolean functions as an efficient tool for proving theorems on Boolean function decompositions.	
Library of Congress Classification (LCC)	5001-6182	Business
	5201-5982	Business Science
	QA 10.3	Boolean Algebra
Journal of Economic Literature (JEL)	M	Business Administration and Business Economics
	M 11 R 4	Production Management Transportation Systems
	C 69	Mathematical Methods and Programming: Other
European Business Schools Library Group (EBSLG)	85 A	Business General
	260 K 240 B	Logistics Information Systems Management
	250 A	Mathematics
Gemeenschappelijke Onderwerpsontsluiting (GOO)		
Classification GOO	85.00	Bedrijfskunde, Organisatiekunde: algemeen
	85.34 85.20	Logistiek management Bestuurlijke informatie, informatieverzorging
	31.43	Functies van meerdere complexe variabelen
Keywords GOO	Bedrijfskunde / Bedrijfseconomie	
	Bedrijfsprocessen, logistiek, management informatiesystemen	
	Bolean-Functions, Modulaire functies, Algoritmen	
Free keywords	Boolean functions, committees, computational complexity, decomposition algorithm, modular decomposition, modular sets, substitution decomposition	

The algorithmic complexity of modular decomposition

Jan C. Bioch

Dept. of Computer Science, Erasmus University Rotterdam,
P.O. Box 1738, 3000 DR Rotterdam.
bioch@few.eur.nl

Abstract. Modular decomposition is a thoroughly investigated topic in many areas such as switching theory, reliability theory, game theory and graph theory. We propose an $O(mn)$-algorithm for the recognition of a modular set of a monotone Boolean function f with m prime implicants and n variables. Using this result we show that the the computation of the modular closure of a set can be done in time $O(mn^2)$. On the other hand, we prove that the recognition problem for general Boolean functions is NP-complete. Moreover, we introduce the so called generalized Shannon decomposition of a Boolean functions as an efficient tool for proving theorems on Boolean function decompositions.

Keywords: Boolean functions, committees, computational complexity, decomposition algorithm, modular decomposition, modular sets, substitution decomposition

1 Introduction

Substitution decomposition has been thoroughly studied by researchers in many different contexts such as switching theory, game theory, reliability theory, network theory, graph theory and hypergraph theory. Möhring and Radermacher [17, 18] give an excellent survey for the various applications of substitution decomposition and connections with combinatorial optimization. They also present a framework for the algebraic and algorithmic aspects of substitution decomposition for a number of discrete structures. Substitution decomposition (disjunctive and non-disjunctive decomposition) for general Boolean functions and partially defined Boolean functions in switching theory is mainly developed by Ashenhurst, Singer, Curtis and Hu [1, 2, 12, 14, 13]. Recently [8, 7, 15] the complexity of non-disjunctive decompositions of partially defined Boolean functions has been determined for various classes of Boolean functions. Decomposition for monotone Boolean functions has been studied in several contexts: game theory (decomposition of n-person games [24]), reliability theory (decomposition of coherent systems [5]) and set systems (clutters [4]). The concepts decomposition and *modular set* are very basic in many contexts and applications. Not surprisingly, the concept of a modular set is rediscovered several times under various names: bound sets, autonomous sets, closed sets, stable sets, clumps, committees, externally related sets, intervals, nonsimplifiable subnetworks, partitive sets and modules, see [9,17] and references therein. In all these contexts the collection of all modular sets is efficiently represented by the so called *decomposition tree* introduced by Shaply in [24]. In graph theory efficient algorithms are known to compute this tree [9,16,10]. The notion of a module in a graph has been recently generalized to hypergraphs in [6]. A unified treatment of all algorithms (up to 1990) related to modular sets known in game theory, reliability theory and set systems (clutters) is given by Ramamurthy [21]. In this paper we are interested in the algorithmic complexity of the decomposition of Boolean functions. In switching theory this complexity has not been discussed. In this context decompositions are based by evaluating so-called Ashenhurt decomposition charts or by using differential calculus [1,2,12,14,13]. It has been shown in [18,17] that the algorithms for the determination of modular sets is exponential in the number of variables. However, here we will study the complexity of decompositions of Boolean functions given in DNF-form. We will show that for general Boolean functions the problem: recognition of a modular set, is NP-complete. For monotone Boolean functions the situation is different. Various decomposition algorithms (in different contexts) are known. Therefore, we briefly discuss the computational aspects of decomposition of monotone Boolean functions.

Computational aspects

It is proved by Singer [22] that the intersection of two modular sets of a Boolean function with a non-empty intersection is again modular. Therefore, each subset C of variables is contained in a smallest modular set called the *modular closure* of C. The modular closure of a set was first introduced by Billera [4] in the context of clutters. Let f be a monotone function defined on the set A, where $|A| = n$, and let m be the number of prime implicants of f. Then according to Möhring and Radermacher [17] the modular tree can be computed in time $O(n^3 T(m,n))$, where $T(m,n)$ is the complexity of computing the modular closure of a set $C \subseteq A$. The first known algorithm due to Billera [4] is based on computing the dual of f. Although this problem is NP-hard in general, for monotone functions the complexity of the dualization problem is still not known, although this problem is unlikely to be NP-hard, see e.g [3]. An improvement of Billera's algorithm by Ramamurthy and Parthasarathy [19] also based on dualization has a similar complexity. The first polynomial algorithm given by Möhring and Radermacher (1984) reduced the complexity to $T(m,n) = O(m^3 n^4)$. Subsequently, the complexity was further reduced by Ramamurthy and Parthasarathy [19] and Ramamurthy [21] to respectively $T(m,n) = O(m^3 n^2)$ and $T(m,n) = O(m^2 n^2)$. It is known that the determination of the modular closure can be solved by solving $O(n)$ times the following problem:

Problem MOD

Input: A Boolean function f with m prime implicants defined on A, where $|A| = n$ and $C \subseteq A$.

Output: "C is modular" if C is modular. An element $x \in Closure(C) \setminus C$ otherwise.

This paper is organized as follows. After introducing some definitions and concepts in section (2), we introduce in section (3) the very useful concept of 'generalized Shannon decompostion' and we argue that this concept can be used to simplify decomposition theory. In section (4) we discuss the complexity of decomposition for general Boolean functions. Decompositions of monotone Boolean functions are discussed in section (5). In section (6) we prove that problem MOD can be solved in linear time. The last section contains the conclusions and topics for further research.

2 Definitions and notations

A Boolean function $f : \{0,1\}^n \mapsto \{0,1\}$ is called monotone(positive) on $N = \{1,2,\cdots,n\}$, if $x \leq y \Rightarrow f(x) \leq f(y)$. A Boolean function f is called *degenerated* if it is constant: $f \equiv 0$ (denoted by $f = \bot$) or $f \equiv 1$ (denoted by $\top$). Otherwise f is called *non-degenerated.* We frequently abbreviate the notation for a DNF of a function f by identifying the variables with their indices and by using $+$ for $\vee$.

Example 1. The function $f(x_1,x_2,x_3) = x_1\bar{x}_2 \vee x_3$ is denoted by $f = 1\bar{2}+3$.

A variable x_j of f is called *essential* if the *restrictions* respectively defined by:

$$f(x_j = 0) = f(x_1 \cdots x_{j-1}, 0, x_{j+1}, \cdots, x_n) \text{ and}$$
$$f(x_j = 1) = f(x_1 \cdots x_{j-1}, 1, x_{j+1} \cdots, x_n),$$

are not identical. The set of all essential variables is denoted by $E(f)$. The *dual* of the function f is defined by: $f^d(x) = \bar{f}(\bar{x})$. Given a function f in DNF, then the dual is obtained by interchanging $\wedge$ and $\vee$.

Disjunctive decompositions

Let $f :\ \{0,1\}^n \mapsto \{0,1\}$ be a Boolean function and $A = \{1,2,\cdots,n\}$. Identify each $i \in A$ with the variable x_i. Then f is said to be a function defined on A. Furthermore, if $A = A_1 \cup A_2 \cup \cdots, A_n$ is a partition of A ($A_i \cap A_j = \emptyset, i \neq j$), then we will denote this by $x_A = (x_{A_1}, \cdots, x_{A_n})$ and $f(x_A) = f(x_{A_1}, \cdots x_{A_n})$. Let $F(y'_A)$ and $g_i(x_{B_i})$ be Boolean functions defined on the mutually disjoint sets $A' = \{1,\cdots,m\}$ and $B_i,\ i \in A'$, and let $A = \cup_{i=1}^m B_i$. Then the Boolean function defined by

$$f(x_A) = F(g_1(x_{B_1}), \cdots, g_m(x_{B_m})),$$

is called the *composition* of the functions F and $g_i,\ i \in A'$, obtained by *substitution* of the variables y_i in F by the functions $g_i,\ i \in A'$. This composition is denoted by $F[g_i,\ i \in A']$. A composition is called *proper* if $|A'| > 1$ and

$|B_i| > 1$ for some $i \in A'$. A Boolean function is said to be *decomposable* if it has a representation as a proper composition. Otherwise, the function f is called *indecomposable* or *prime.* If $F[g_i,\ i \in A']$ is a decomposition of the function$f$ then the partition $\pi = \{B_i,\ i \in A'\}$ is called a *congruence partition* and F is called the *quotient* of f modulo π and is denoted by f/π. From the definition of decomposition it easily follows that

$$f = F[g_i,\ i \in A'] \Leftrightarrow f^d = F^d[{g_i}^d,\ i \in A'].$$

Therefore, we have $F = f/\pi \Leftrightarrow F^d = f^d/\pi$. Moreover, it is well-known that the functions $g_i,\ i \in A'$, are determined modulo complementation of the functions, and that the quotient F is determined modulo complementation of the variables. The algebraic properties of congruence partitions are discussed in [17, 18]. It is known that each decomposition of a Boolean function f can be obtained by a series of so called *simple disjunctive decompositions.* These are decompositions of the form

$$f(x_A) = F(x_B, g(x_C)),$$

where $\pi = \{B, C\}$ is a partition of A.

Definition 1. *Let f be a Boolean function defined on A Then $C \subseteq A$ is called a modular set of f if f has a simple disjunctive decomposition of the form $f(x_A) = F(x_B, g(x_C))$. The function g is called a component of f.*

The following theorem is fundamental:

Theorem 1. *Let f be a general Boolean function. Suppose A and B are incomparable modular sets such that $A \cap B \neq \emptyset$. Then $A\bar{B}, A \cap B, \bar{A}B$ and $A \cup B$ are modular sets of f, and $f(x_{A\cup B}) = f(x_{A\bar{B}}) \circ f(x_{A\cap B}) \circ f(x_{\bar{A}B})$, where $\circ$ is either $\wedge$, $\vee$ or $\oplus$. If f is monotone, then $\circ$ is either $\wedge$ or $\vee$.*

Remark 1. Theorem (1) is a famous result called the *Three Modules Theorem* of Ashenhurst [2], reproved in game theory and reliability theory [21]. But as far as we know this result is due to Singer [22].

Example 2. Let f be the monotone function defined by $f = 134+234+135+235$. Let $A = \{1, 2, 3\}$, and $B = \{3, 4, 5\}$. Then A, B and $A \cap B$ are modular and $f = (1 + 2)3(4 + 5)$.

Definition 2. *Let f be a Boolean function defined on A. The closure of $C \subseteq A$ is defined by: $Cl_{(f)}(C) = \cap\{B \mid C \subseteq B,\ B$ is a modular set of $f\}$.*

3 Generalized Shannon decomposition

Let f be a Boolean function on A. Then for all $j \in A$ the following decomposition holds:

$$f = \bar{x}_j f_{x_j=0} \vee x_j f_{x_j=1}. \qquad (1)$$

Equation (1) is known as a *Shannon decomposition* of f. Now consider the simple disjunctive decomposition

$$f(x_A) = F(x_B, g(x_C)). \tag{2}$$

Then using equation (1) we have:

$$f(x_A) = \bar{g}(x_C)F_0(x_B) \vee g(x_C)F_1(x_B), \tag{3}$$

where $F_0(x_B) = F(x_B, 0)$ and $F_1(x_B) = F(x_B, 1)$.
Conversely, let g and h_0, h_1 be arbitrary Boolean functions defined respectively on C and B such that $f = \bar{g}h_0 \vee gh_1$, and let the function F be defined by $F(x_B, y) := \bar{y}h_0 \vee yh_1$. Then $f(x_A) = F(x_B, g(x_C))$ is a simple disjunctive decomposition of f, where $F_0(x_B) = h_0$ and $F_1(x_B) = h_1$. Therefore, we have proved the following fundamental lemma:

Lemma 1. *Let f be a Boolean function on A. Then $C \subseteq A$ is a modular set of f iff there exists a Boolean function g on A and functions h_0 and h_1 on $B = A \setminus C$ such that $f = \bar{g}h_0 \vee gh_1$.*

We call the decomposition in the previous lemma *a generalized Shannon decomposition*. In particular, we call the decomposition in equation (3) a generalized Shannon decomposition associated with the simple disjunctive decomposition (2). If C is a modular set of the function f such that C contains at least one essential variable of f, then it follows from the decomposition

$$f = \bar{g}h_0 \vee gh_1, \tag{4}$$

that the function g is non-degenerate and that the functions h_0 and h_1 are not identical. Therefore, there exists a binary vector b_0 such that either $g(x_C) = f(b_0, x_C)$ or $\bar{g} = f(b_0, x_C)$. This shows that we may assume that the function g is a subfunction of f.

Definition 3. *Let C be a modular set of f. Then a non-constant subfunction $f(b_0, x_C)$ is denoted by $f_C(x_C)$. For general Boolean functions this subfunction is determined modulo complementation. For monotone Boolean functions the function $f_C(x_C)$ is uniquely determined and called the contraction ([21]) of f wrt to C.*

In general, equation (4) shows that if b is a fixed vector then the function $f(b, x_C)$ is either degenerate or identical to g of identical to $\bar{g}$. It is not difficult to see that the converse holds also. Therefore, the following theorem holds:

Theorem 2. *Let f be a Boolean function defined on A. If $C \subseteq A$ contains at least one essential variable of f, then the following statements are equivalent:*

a) *C is modular*
b) *There exists a vector b_0 such that the function $g(x_C) := f(b_0, x_C)$ is non-degenerate and for all fixed b the function $f_b := f(b, x_C)$ is degenerate or identical to either g or $\bar{g}$.*

Lemma 2. *Let f be a Boolean function defined on V and let $\{A, B, C, D\}$ be a partition of V. Suppose $f(x_V) = F(g(x_{A\cup B}), x_C, x_D) = G(x_A, h(x_{B\cup C}), x_D)$. Then there exist functions H and k such that $f(x_V) = H(g(x_A, k(x_B), x_C, x_D))$.*

Proof. In the proof the variables in D do not play a role. Therefore, we will assume in our notation that f is actually defined on the partition $\{A, B, C\}$. From theorem (1) it follows that $g = f_{A\cup B}$. Therefore, there exists a vector c such that:

$$g(x_A, x_B) = G(x_A, h(x_B, c)) = \bar{k}(x_B)G_0(x_A) \vee k(x_B)G_1(x_A), \tag{5}$$

where k is defined as: $k(x_B) = h(x_B, c)$. According to equation (4), we also have:

$$f(x_V) = \bar{g}(x_A, x_B)F_0(x_C) \vee g(x_A, x_B)F_1(x_C). \tag{6}$$

Combining 5 and 6 gives after some re-grouping of terms:

$$f(x_V) = H_0(x_{A\cup C})\bar{k}(x_B) \vee H_1(x_{A\cup C})k(x_B) = H(x_A, k(x_B), x_C).$$

□

It is easy to see that lemma (2) is equivalent to the following theorem due to Singer [22]:

Theorem 3. *Let f be a Boolean function defined on N. If $A, B \subseteq N$ are modular sets of f such that $A \cap B \neq \emptyset$, then $A \cap B$ is also a modular set of f.*

Remark 2. This fundamental theorem is proved in the literature in a much more elaborate way by considering Ashenhurst decomposition charts, expansions of Boolean functions or differential calculus ([1, 2, 12, 14, 13]). In fact the theory using Ashenhurt decomposition charts can be more easily developed by using the concept 'generalized Shannon decomposition' discussed in this section.

4 Complexity of decomposition for general Boolean functions

In this section we prove that for general Boolean functions the problem of recognizing modular sets (called MODULAR) is coNP-complete.

Problem MODULAR
Given: A Boolean function f in DNF defined on A and a set $C \subset A$ that contains at least one essential variable of f.
Question: Is C a modular set of f ?

We relate this problem to the following recognition problem that is coNP-complete:

Problem COMPLEMENT
Given: Boolean functions f and g in DNF.
Question: $f = \bar{g}$?

Lemma 3. *Problem COMPLEMENT is reducible to MODULAR*

Proof. We will prove this lemma by reducing problem COMPLEMENT to problem MODULAR. Suppose g_1 and g_2 are Boolean functions given in DNF on $C = \{x_1 \cdots, x_n\}$. Define the function f on $C \cup \{x, y\}$ as:

$$f = xg_1 \vee yg_2. \tag{7}$$

If $g_2 = \bar{g}_1$, then C is a modular set of f. Conversely, suppose C is modular and C contains essential variables of f. Then there exists a pair of binary values (x_0, y_0) such that the function g defined by $g = f(x_0, y_0, x_C)$ is non-degenerate. Furthermore, according to theorem (2) for all *fixed* x and y the function $h(x_C) = f(x, y, x_C)$ is constant or identical to the function g or its complement. From equation (7) it follows that $h \in \{\perp, g_2, g_1, g_1 \vee g_2\}$. Therefore, we have $g_2 = \bar{g_1}$. Conclusion: $g_2 = \bar{g}_1 \Leftrightarrow C$ is modular.

□

The following lemma is easy to prove:

Lemma 4. *Suppose $C \subseteq N$ and g and f^c are Boolean functions defined on respectively $\bar{C}$ and N. Let f be function defined by $f = g \vee f^c$. Then C is a modular set of f iff C is a modular set of f^c.*

Lemma 5. *MODULAR is in coNP.*

Proof. Let f be a Boolean function on A and C a subset of A that contains at least one essential variable of f. Suppose $f = g \vee f^c$ (see lemma (4)) and $\bigvee_{j=1}^{m} c_j$ is a DNF-representation of f^c, where we assume that each term c_j contains a variable in C. We may assume also that each term c_j contains a variable in $\bar{C}$, for otherwise we replace c_j by $c_j = x_0 c_j \vee \bar{x}_0 c_j$, where $x_0 \in C$. Therefore, each term c_j can be written as $c_j = s_j t_j$, where s_j and t_j are conjunctions defined on $\bar{C}$ respectively C. Let $S = \{s_i \mid i \in \bar{C}\}$ and $T = \{t_j \mid j \in C\}$. Finally, let $\phi = \bigvee_{i \in S} s_i$ and $\psi = \bigvee_{j \in T} t_j$. We now define the *connection-matrix* $A = (\alpha_{i,j})$ by $\alpha_{i,j} = 1 \Leftrightarrow s_i t_j$ is a term of f^c $(i \in S, j \in T)$. By construction each row and column of A has a non-zero entry. If A is constant, then $f^c = \phi \wedge \psi$, implying that C is modular. Otherwise, the matrix A contains at least one non-constant row R_p. To check the modularity of C we define the functions $g = \bigvee_j \{s_p t_j \mid \alpha_{p,j} = 1\}$, $h_0 = \bigvee \{s_i t_p \mid \alpha_{i,k} = 1\}$, where k is a fixed index such that $\alpha_{p,k} = 0$, and $h_1 = \bigvee_i s_i t_l \mid \alpha_{i,l} = 1\}$, where l is a fixed index such that $\alpha_{p,l} = 1$. Then according to theorem (4) C is non-modular iff $f \neq h_0 \bar{g} \vee h_1 g$. All the constructions in the proof can be done in time $\mathrm{O}(n^2 m^2)$. Moreover, to show that C is not modular, it is sufficient to exhibit a binary vector x such that $f(x) \neq h_0(x_{A \setminus C}) \bar{g}(x_C) \vee h_1(x_{A \setminus C}) g(x_C)$. This establishes that problem MODULAR is in coNP.

□

The lemma's (3) and (5)imply:

Theorem 4. *Problem MODULAR is coNP-complete.*

5 Decompositions of monotone Boolean functions

In this section we will frequently represent a subset $C \subseteq N$ by its characteristic vector $c \in \{0,1\}^n$. A positive Boolean function has a unique irredundant DNF consisting of all prime implicants. The set of prime implicants correspond to the this of minimal true vectors of f, denoted by $\min T(f)$. It is well-known that $\min T(f^d)$ represents the set of minimal transversals of $\min T(f)$. The complement of a false vector is a transversal: $f(x) = 0 \Leftrightarrow f^d(\bar{x}) = 1$.

Example 3. Let f be the function defined by $f(x) = x_1 x_2 \vee x_2 x_3$ Then:
$f^d(x) = (x_1 \wedge x_3)(x_2 \wedge x_3) = x_2 \wedge x_1 x_3$, $\min T(f^d) = \{010, 101\}$ are the minimal transversals of $\min T(f) = \{110, 011\}$, and 001 is a false vector and its complement 110 is a transversal of $\min T(f)$.

Definition 4. *For a monotone function f the function f^c is defined by: $minT(f^c) = \{v \mid v \in minT(f), v \wedge c \neq 0\}$.*

From this definition it follows that every monotone Boolean function f has the following *basic* decomposition:

$$f = f(c = 0) \vee f^c. \tag{8}$$

Furthermore for a monotone Boolean function f Shannon's decomposition has the form:

$$f(x) = f(x_j = 0) \vee x_j f(x_j = 1). \tag{9}$$

Definition 5. *Let f be a monotone function. Then the contraction of f on c is defined by $f_c = f^c(\bar{c} = 1)$, where $(\bar{c} = 1)$ indicates that all the variables in $\bar{C}$ are replaced by 1.*

The following characterization of the contraction is well known, see [21]:

Theorem 5. *Let $x \leq c$. Then: $f_c(x) = 1 \Leftrightarrow \exists y \leq \bar{c}$ such that $f(y) = 0$ and $f(x \vee y) = 1$.*

Example 4. Let the monotone function f be defined by:
$f = 1245 + 126 + 2345 + 236 + 46$ and let $C = \{1, 2, 3\}$. Then
$c = 111000$, and $f(c = 0) = 46$, $f^c = 1245 + 126 + 2345 + 236$, $f_c = 12 + 23$.

Theorem 6. *Let f be a monotone Boolean function defined on N and let $C \subseteq N$. Then C is modular iff*

$$f = f(c = 0) \vee f^c(c = 1) f_c \Leftrightarrow \mathbf{f^c} = \mathbf{f^c}(\mathbf{c} = \mathbf{1})\mathbf{f_c}.$$

Proof. If C is modular, then $f = F(x_B, g(x_C))$, where $\{B, C\}$ is a partition of N. Then Shannon's decomposition: $F(x_B, y) = F(y = 0) \vee y F(y = 1)$, implies:

$$f = f(c = 0) \vee g f(c = 1). \tag{10}$$

Furthermore, since $f(c=1) = f^c(c=1) \vee f(c=0)$ by equation (8), equation (10) implies:

$$f = f(c=0) \vee gf^c(c=1). \tag{11}$$

Using the fact that the functions $f(c=0)$ and $f^c(c=1)$ are defined on $B = N \setminus C$, equation (11) implies:

$$f_c(x_C) = f^c(\bar{c}=1)(x_C) = g(x_C).$$

Therefore, we have the decomposition:

$$f = f(c=0) \vee f^c(c=1)f_c. \tag{12}$$

Conversely, if equation (12) holds, then C is modular.

□

Remark 3. Note, that by checking the equation $f^c = f^c(c=1)f_c$ the problem of deciding whether a set C is modular or not can be solved in time $O(m^2n^2)$!

Example 5. Consider the function f of the previous example, and let $C = \{1,2,3\}$. Then: $f^c = f^c(c=1)f_c = (45+6)(12+23)$.

Characterizations of modular sets

Its is known [21] that a modular set can be characterized as follows:

Theorem 7. *Suppose f is a monotone, non-degenerate function defined on A, $C \subseteq A$ and $e(f) \wedge c \neq \mathbf{0}$, where $e(f)$ denotes the characteristic vector of the set of essential variables of f. Then the following are equivalent:*

a) *C is a modular set of f*
b) $(f^d)_c = (f_c)^d$
c) *C is a modular set of f^c*
d) $\forall \mathbf{v}, \mathbf{w} \in \min \mathbf{T}(\mathbf{f^c}) : \mathbf{f}(\mathbf{vc} \vee \mathbf{w\bar{c}}) = \mathbf{1}$
e) $\min T(f^c) = \{vc \vee w\bar{c} \mid v, w \in \min T(f^c)\}$
f) $e(((f^c)^d)^c) = e(f) \wedge c.$

Example 6. Consider the function $f = (12+23)(45+6) = 1245+126+2345+236$. If C $= \{1,2\}$ or $C = \{1,2,3\}$, then $f^{cd} = 2+13+46+56$. If $C = \{1,2,3\}$, then $e(f^{cdc}) = c$. However, if $C = \{1,2\}$, then C is not modular because $\exists v \in \min T(f^{cd})$ with $v \wedge c \neq \mathbf{0}$ such that $v \not\leq c$.

Computing the modular closure

The following theorem [21] relates the modular closure of f^c to its dual:

Theorem 8. $c \leq e(f^{cdc}) \leq Cl_{f^c}(c) \leq Cl_f(c)$.

Theorem 9. *Suppose f is a monotone function and $u, v \in minT(f^c)$. If $f(uc \vee v\bar{c}) = 0$, then the vector $t = \bar{u}c \vee \bar{v}\bar{c} \in T(f^{cd})$. Furthermore, $\forall w \in minT(f^{cd})$ such that $w \leq t$ we have $\mathbf{0} \not\leq w\bar{c} \leq e(f^{cdc})$.*

Proof. It is easy to see that $\bar{t} = uc \vee v\bar{c}$, so $t \in T(f^{cd})$. Furthermore, the assumptions imply $w \leq \bar{u}c \vee \bar{v}\bar{c}$, and $\bar{u}, \bar{v} \in F(f^{cd})$. Therefore, since $w \in minT(f^{cd})$ we conclude $w \not\leq \bar{u}c$ and $w \not\leq \bar{v}\bar{c}$, implying $w\bar{u}c \neq \mathbf{0}$ and $w\bar{v}\bar{c} \neq \mathbf{0}$. From this we conclude that $w \leq e(f^{cdc})$ and that $w\bar{c} \neq \mathbf{0}$.

□

Remark 4. Given t, then a vector w in theorem (9) can be determined in time $O(mn^2)$, since it is known that a minimal transversal w can be obtained from a transversal t in $O(n)$ steps. Therefore, the theorem shows that we can determine an element in $Cl_f(C) \setminus C$ from t in time $O(mn^2)$.

Definition 6. *Suppose $\exists u, v \in \min T(f^c)$ such that $f(uc \vee v\bar{c}) = 0$. Then we call the vector $uc \vee v\bar{c}$ a culprit of f wrt c.*

The following lemma is of independent interest:

Lemma 6. *Suppose f is a monotone Boolean function and $f^d(w) = 1$. Let $v \in argmin\{|uw| \mid u \in minT(f)\}$. Then for all unit vectors $e \leq wv$ there exits a vector $w_0 \in minT(f^d)$ such that $e \leq w_0 \leq w$.*

Proof. Since $w\bar{v} \wedge v = \mathbf{0}$ and $v \in minT(f)$ we conclude that $w\bar{v} \notin T(f^d)$. On the other hand we claim that

$$w\bar{v} \vee e \in T(f^d). \tag{13}$$

To prove this claim we suppose that $u \in minT(f)$ but $(w\bar{v} \vee e) \wedge u = \mathbf{0}$. Then we have $e \not\leq u$ and $w\bar{v}u = \mathbf{0}$. However, the last equality implies $wu \leq v$, implying

$$\mathbf{0} \neq wu \leq wv. \tag{14}$$

By the minimality assumption we then have $wu = wv$. Since $e \not\leq u$ and $e \leq wv$, this is a contradiction. This proves our claim (13). Furthermore we claim that:

$$\forall w_0 \in minT(f^d) \text{ such that: } w_0 \leq w\bar{v} \vee e, \text{ we have } e \leq w_0. \tag{15}$$

To prove claim (15), assume $e \not\leq w_0$. Then we would have: $w_0 \leq w\bar{v}$. However, $w\bar{v} \notin T(f^d)$, so $w_0 \not\leq w\bar{v}$. Contradiction. This finishes our proof.

□

A useful variation of this lemma is:

Lemma 7. *Suppose f is a monotone Boolean function and $f^d(w) = 1$. Let c be a vector such that $U = \{u \in minT(f) \mid uwc = \mathbf{0}\} \neq \emptyset$. Let $v \in argmin_{u \in U}\{\ |uw|\ \}$. Then for all unit vectors $e \leq wv$ there exits a vector $w_0 \in minT(f^d)$ such that $e \leq w_0 \leq w$.*

Proof. Note, that the inequality (14) implies: $wuc \leq wvc = \mathbf{0}$, so $u \in U$. Using this observation the proof of this lemma is the same as the proof of lemma (6)

□

The following fundamental theorem is a variation of a theorem in [21]:

Theorem 10. *Let f be a monotone function. Suppose t is the complement of a culprit of f wrt to c. Then $U = \{u \in minT(f^c) \mid utc = \mathbf{0}\} \neq \emptyset$. Furthermore, if $u_0 \in argmin_{u \in U}\{\ |ut|\ \}$, then $\mathbf{0} \neq u_0 t = u_0 t\bar{c} \leq Cl_f(c)$.*

Proof. Since t is the complement of a culprit we have $\exists v, w \in minT(f^c)$ such that $t = \bar{v}c \vee \bar{w}\bar{c}$, and $f^{cd}(t) = 1$. Furthermore, since $\bar{v}c \notin T(f^{cd})$ there must exist a vector $u_0 \in minT(f^c)$ such that $u_0\bar{v}c = \mathbf{0}$. From $u_0 tc = u_0\bar{v}c = \mathbf{0}$ it follows that $u_0 \in U$. Now suppose $u_0 \in argmin_{u \in U}\{\ |ut|\ \}$, then according to lemma (7): for all unit vectors $e \leq u_0 t$ we have: $\exists t_0 \in minT(f^{cd})$ such that $e \leq t_0 \leq t$. Now theorem (8) implies $\mathbf{0} \neq t_0\bar{c} \leq e(f^{cdc})$. Therefore, we have: $\mathbf{0} \neq u_0 t = u_0 t\bar{c} \leq Cl_f(c)$.

□

Remark 5. The vector $u_0 t$ can be determined in $O(mn)$ time. Therefore, if a culprit is known, then we can determine in $O(mn)$ time an element in $Cl_f(C) \backslash C$.

6 Solving MOD in linear time

In this section we show that the problem MOD introduced in section (1) can be solved in linear time: $O(mn)$. We first introduce some notations for a given monotone Boolean function f on A: $M = \min T(f^c) = \{v_1, \cdots, v_m\}$, $S = \{vc \mid v \in M\}$ and $T = \{v\bar{c} \mid v \in M\}$. Let $p = |S|$ and $q = |T|$. For each $v \in M$ we can write $v = vc \vee \bar{v}c$ as a $2n$-vector: $(vc|v\bar{c})$. Now we consider the list of all (column-)vectors:

$$\begin{array}{c|c|} vc & v_1c\ v_2c \cdots\cdots v_mc \\ v\bar{c} & v_1\bar{c}\ v_2\bar{c} \cdots\cdots v_m\bar{c} \end{array}\ .$$

According to [23], the set of all these $2n$-vectors can be lexicographical sorted in time $O(mn)$. Now it follows from the next lemma (8) that C is modular iff the sorted list of all $2n$-vectors has the following structure:

$$\mathcal{S} = \begin{array}{c|c|c|c|c|} vc & s_1 \cdots s_1 & s_2 \cdots s_2 & \cdots\cdots & s_p \cdots s_p \\ v\bar{c} & t_1 \cdots t_q & t_1 \cdots t_q & \cdots\cdots & t_1 \cdots t_q \end{array}\ .$$

So if C is modular, then the structure $\mathcal{S}$ consists of p segments of length q, and $m = pq$. It is easy to see by scanning from left to right that the structure $\mathcal{S}$ can be identified in time $O(mn)$. Therefore, it can be determined in time $O(mn)$ whether a set C is modular or not. However, the more difficult part is to show that we can find an element $x \in Closure(C) \setminus C$ in linear time if C is not modular. We call such an element a *culprit* wrt the non-modularity of C.

Finding a culprit in linear time

Let V, W be subsets of A, and let v and w denote their characteristic vectors. Then we denote $V < W$ respectively $V > W$ by $v < w$ and $v > w$. Furthermore, we use the following notations: $v \sim w \Leftrightarrow (v < w \text{ or } v > w)$, and $v \simeq w \Leftrightarrow (v \leq w \text{ or } v > w)$. The next basic lemma is used several times in order to find a culprit.

Lemma 8. *Let s_1, s_2 and t_1, t_2 denote arbitrary elements in respectively the first and second row of the list S. Then:*

a) $s_1 \simeq s_2 \Rightarrow t_1 \not\simeq t_2$
b) $t_1 \simeq t_2 \Rightarrow s_1 \not\simeq s_2$
c) *If either $s_1 \sim s_2$ or $t_1 \sim t_2$, then either $s_1 \vee t_2$ or $s_2 \vee t_1$ is a culprit.*
d) *If $s_1 \vee t_2$ does not occur in the list S and s_1 and t_2 are minimal, then $s_1 \vee t_2$ is a culprit.*

Proof. c) Let v and w be minimal vectors of f^c such that $s_1 = vc, s_2 = wc, t_1 = v\bar{c}$ and $t_2 = w\bar{c}$. Suppose $s_1 \sim s_2$, e.g $vc > wc$. Then $v = vc \vee v\bar{c} > wc \vee v\bar{c}$. Since v is a minimal vector of f^c, the vector $wc \vee v\bar{c}$ is a culprit: $f(wc \vee v\bar{c}) = 0$, see theorem (7 c). The other assertions are proved similar.

□

Corollary 1. *If $s_1 \vee t_2$ does not occur in the list S, than a culprit can be found in time $O(mn)$, see the next example (7).*

We will now describe our algorithm to decide if a set C is modular, or otherwise to find a culprit. The overall algorithm is given in the procedure Modular.

```
Modular($L$, **var** $culprit$):
    $flag := false; culprit := false$
    call FirstSegment
    **while** $flag = true$ **do** call NextSegment
```

The procedure Firstsegment scans the list $\mathcal{S}$ from left to right, by comparing each element in the first row by s_1. In this procedure we determine the length of the first segment and the first element in the next segment. While there is a next segment, i.e if there is an element $s_i \neq S_1$ indicated by $flag = true$, then we start the procedure Nextsegment. Both procedures determine the beginning of the next segment by updating the variable *index*. The beginning of each next segment is given by S_1.

```
FirstSegment($L$, **var** $index, flag, culprit, p, q$):
    **if** $s_1 \neq s_2$ **then**
        **if** $s_1 \sim s_2$ **then** return $culprit$
            **else if** $\forall j > 1\ t_j = t_1$ **then** return $(q = 1, p = m)$
                **else** $j_0 := \min\{j \mid t_j \neq t_1\}$
                (so $s_1 \vee t_{j_0}$ is not in $L$) return $culprit$
        **else if** $\forall i > 2\ s_i = s_1$ **then** return $(p = 1; q = m)$
            **else** $i_0 := \min\{i \mid s_i \neq s_1\}$;
            **if** $s_{i_0} \sim s_1$ **then** return $culprit$
                **else** return $(q = i_0 - 1, p = m/q, index = q + 1, flag = true)$
```

The procedure Nexsegment also detects whether the length of each next segment is equal to p. If not, then either $S_1 \vee t_i$ or $s_i \vee T_1$ is not in the list. In that case we scan the list to find an element s_i or t_j comparable to respectively S_1 and T_1, see corollary (1)

Example 7. Let $f = 15 + 16 + 245 + 35 + 36 + 46$, and $C = \{1, 2, 3, 4\}$. Then the sorted list is given by $\mathcal{S} = \begin{array}{cc|cccc} 1 & 1 & 24 & 3 & 3 & 4 \\ 5 & 6 & 5 & 5 & 6 & 6 \end{array}$. In this example the first segment has length $p = 2$. Since the fourth element in the first row is not equal to 24 we detect that 246 is not in $\mathcal{S}$. By comparing 24 with the next elements in the first row we discover that 4 is comparable with 24. Hence 45 is not a true vector of f^c. Therefore the vector 000110 is a culprit.

□

```
NextSegment($L$, **var** $index, flag, culprit$):
    $flag := false; i := 2; S_1 := s_{index}$
    **while** $S_i = S_1$ **do** $i := i + 1$
    $\acute{q} := i - 1$
    **if** $\acute{q} \neq q$ **then** (note: either $S_1 \vee T_i$ or $S_i \vee T_1$ is not in $L$) return $culprit$
        **else** call Compare
    **if** $S_{q+1} \sim S_1$ **then** return $culprit$
        **else** return $(flag = true, index = q + 1)$
```

Even if all the elements in the first row of a segment are equal to those of the first segment, we have to compare all the elements of the second row with those of the elements of the first segment in the second row. This comparison is made in the procedure Compare called in the procedure Nextsegment.

```
Compare($T$, **var** $culprit$):
    $culprit := false$
    **if** $\forall j \in \{1, \cdots q\}\ T_j = t_j$ **then** return
        **else** $j_0 := \min\{j \mid T_j \neq t_j\}$
    **if** $T_{j_0} \sim t_{j_0}$ **then** return $culprit$
        **else** $(s_1 \vee T_{j_0}$ or $S_{j_0} \vee t_1$ is not in $L)$ return $culprit$
```

7 Conclusions and further research

For monotone Boolean functions the recognition of modular sets and therefore the computation of the modular closure and the modular tree can be reduced with a factor $O(m)$. On the other hand we have proved that for general Boolean functions the recognition problem is NP-complete. We also argued that the generalized Shannon representation of a disjunctive decomposition is an effective tool to study decompositions of Boolean functions. Compared with the set theoretic approach used in the literature it appears that the Boolean function approach is more transparent. Since partially defined Boolean functions play an important role in many datamining tasks we consider decomposition theory in datamining also as an important task for further research. Finally decompositions with components restricted to a certain class, e.g. self-dual functions (committees in game theory), regular functions etc. are an interesting topic for future research.

Acknowledgement: I would like to thank Peter Hammer, Endre Boros and Yves Crama for some stimulating discussions and pointers to the literature.

References

1. Ashenhurst, R.L. (1952): The Decomposition of Switching Functions. Bell Laboratories' Report No. BL-1(II) (reprinted in [12])
2. Ashenhurst, R.L. (1959): The decomposition of switching functions. Proc. International Symposium on the Theory of Switching, Part I (vol. XXIX, Ann. Computation Lab. Harward), Harward University Press, Cambridge, 75–116
3. Bioch, J.C., Ibaraki, T. (1995): Complexity of Identification and Dualization of Positive Boolean Functions. Information and Computation 123, 50–63
4. Billera, L.J. (1970): On the composition and decomposition of clutters. Journal of Combinatorial Theory 11, 234–245
5. Birnbaum, Z.W., Essary, J.D. (1965): Modules of coherent systems. SIAM Journal of Applied Mathematics 13, 444–462
6. Bonizzoni, P., Vedova G.D. (1999): An algorithm for the Modular Decomposition of Hypergraphs. Journal of Algorithms 32, 65–86
7. Boros, E., Gurvich, V., Hammer, P.L., Ibaraki, T., Kogan, A. (1994): Decomposition of partially defined Boolean functions. Dimacs Technical Report 94-09, Rutgers University, Rutcor Research Report, RRR-13-94, Discrete Applied Math., 62, 51–75
8. Boros, E., Gurvich, V., Hammer, P.L., Ibaraki, T., Kogan, A. (2001): Structural analysis of partially defined Boolean functions. Annals of Operations Research. Vol. Optimal partitioning of combinatorial structures.
9. Brandt, Le, V.B., Spinrad, J.P. (1999): Graph Classes: A Survey. SIAM Monographs on Discrete Mathematics and Applications
10. Cournier, A., and Habib M. (1994): A new linear algorithm for modular decomposition, LNCS, vol. 787, 68-84, Springer-Verlag, Berlin
11. Butterworth, R.W. (1972): A set theoretic treatment of coherent systems. SIAM Journal of Applied Mathematics 22, 590–598
12. Curtis, H.A. (1962): A New Approach to the Design of Switching Circuits. Van Nostrand, Princeton
13. Davio, M., Deschamps, J.P., Thayse, A. (1978): Discrete and Switching Functions, McGraw-Hill
14. Hu, S.T.(1968): Mathematical theory of switching circuits and automata, University of California Press, Berkely and Los Angeles
15. Makino, K., Yano, K., Ibaraki, T. (1995) Positive and Horn Decomposibility of Partially Defined Boolean Functions, Rutgers University, Rutcor Research Report, RRR-29-95.
16. McConnell, R.M., Spinrad J.P. (1996): Linear-time modular decomposition of undirected graphs and efficient orientation of comparabitlity graphs, Proc. of the 5th ACM-SIAM Symposium on Discrete Algorithms, SODA'96, 536-545
17. Möhring, R.H., Radermachter, F.J. (1984): Substitution decomposition of discrete structures and connections to combinatorial optimization. Annals of Discrete Mathematics, 19, 257–356
18. Möhring, R.H. (1985/86): Algorithmic aspects of the substitution of decomposition in optimization over relations, set systems and Boolean Functions. Annals of Operations Research 4, 195–225
19. Ramamurthy, K.G., Parthasarathy, T. (1986): An algorithm to find the smallest committee containing a given set. Opsearch, 23, 1–6
20. Ramamurthy, K.G. (1988): A new algorithm to find the smallest committee containing a given set of players. Opsearch 25, 49–56

21. Ramamurthy, K.G. (1990): Coherent Structures and Simple Games. Theory and Decision Library, Series C, Kluwer, Dordrecht
22. Singer, T. (1953): The Decomposition Chart as a Theoretical Aid. Bell Laboratories' Report No., III-1-III-28, 1953. (reprinted in [12])
23. Sedgewick R. (1990): Algorithms in C. Addison-Wesley
24. Shapley, L.S. (1967): On committees. In New Methods of Thought and Procedure, edited by F. Zwicky and A.G. Wilson, Springer-Verlag, New York, 246–270

Publications in the Report Series Research[*] in Management

ERIM Research Program: "Business Processes, Logistics and Information Systems"

2001

Bankruptcy Prediction with Rough Sets
Jan C. Bioch & Viara Popova
ERS-2001-11-LIS

Neural Networks for Target Selection in Direct Marketing
Rob Potharst, Uzay Kaymak & Wim Pijls
ERS-2001-14-LIS

An Inventory Model with Dependent Product Demands and Returns
Gudrun P. Kiesmüller & Erwin van der Laan
ERS-2001-16-LIS

Weighted Constraints in Fuzzy Optimization
U. Kaymak & J.M. Sousa
ERS-2001-19-LIS

Minimum Vehicle Fleet Size at a Container Terminal
Iris F.A. Vis, René de Koster & Martin W.P. Savelsbergh
ERS-2001-24-LIS

The algorithmic complexity of modular decompostion
Jan C. Bioch
ERS-2001-30-LIS

A Dynamic Approach to Vehicle Scheduling
Dennis Huisman, Richard Freling & Albert Wagelmans
ERS-2001- 35-LIS

Effective Algorithms for Integrated Scheduling of Handling Equipment at Automated Container Terminals
Patrick J.M. Meersmans & Albert Wagelmans
ERS-2001-36-LIS

Rostering at a Dutch Security Firm
Richard Freling, Nanda Piersma, Albert P.M. Wagelmans & Arjen van de Wetering
ERS-2001-37-LIS

Probabilistic and Statistical Fuzzy Set Foundations of Competitive Exception Learning
J. van den Berg, W.M. van den Bergh, U. Kaymak
ERS-2001-40-LIS

[*] A complete overview of the ERIM Report Series Research in Management:
http://www.ers.erim.eur.nl

ERIM Research Programs:

LIS	Business Processes, Logistics and Information Systems
ORG	Organizing for Performance
MKT	Marketing
F&A	Finance and Accounting
STR	Strategy and Entrepreneurship

2000

A Greedy Heuristic for a Three-Level Multi-Period Single-Sourcing Problem
H. Edwin Romeijn & Dolores Romero Morales
ERS-2000-04-LIS

Integer Constraints for Train Series Connections
Rob A. Zuidwijk & Leo G. Kroon
ERS-2000-05-LIS

Competitive Exception Learning Using Fuzzy Frequency Distribution
W-M. van den Bergh & J. van den Berg
ERS-2000-06-LIS

Models and Algorithms for Integration of Vehicle and Crew Scheduling
Richard Freling, Dennis Huisman & Albert P.M. Wagelmans
ERS-2000-14-LIS

Managing Knowledge in a Distributed Decision Making Context: The Way Forward for Decision Support Systems
Sajda Qureshi & Vlatka Hlupic
ERS-2000-16-LIS

Adaptiveness in Virtual Teams: Organisational Challenges and Research Direction
Sajda Qureshi & Doug Vogel
ERS-2000-20-LIS

Assessment of Sustainable Development: a Novel Approach using Fuzzy Set Theory
A.M.G. Cornelissen, J. van den Berg, W.J. Koops, M. Grossman & H.M.J. Udo
ERS-2000-23-LIS

Applying an Integrated Approach to Vehicle and Crew Scheduling in Practice
Richard Freling, Dennis Huisman & Albert P.M. Wagelmans
ERS-2000-31-LIS

An NPV and AC analysis of a stochastic inventory system with joint manufacturing and remanufacturing
Erwin van der Laan
ERS-2000-38-LIS

Generalizing Refinement Operators to Learn Prenex Conjunctive Normal Forms
Shan-Hwei Nienhuys-Cheng, Wim Van Laer, Jan Ramon & Luc De Raedt
ERS-2000-39-LIS

Classification and Target Group Selection bases upon Frequent Patterns
Wim Pijls & Rob Potharst
ERS-2000-40-LIS

Average Costs versus Net Present Value: a Comparison for Multi-Source Inventory Models
Erwin van der Laan & Ruud Teunter
ERS-2000-47-LIS

Fuzzy Modeling of Client Preference in Data-Rich Marketing Environments
Magne Setnes & Uzay Kaymak
ERS-2000-49-LIS

Extended Fuzzy Clustering Algorithms
Uzay Kaymak & Magne Setnes
ERS-2000-51-LIS