On the P vs NP question: a proof on inequality

Original
On the P vs NP question: a proof on inequality / Meo, Angelo Raffaele. - ELETTRONICO. - (2016), pp. 1-34.

Availability:
This version is available at: 11583/2645861 since: 2016-08-02T12:32:03Z

Publisher:

Published
DOI:10.6092/polito/porto/2645861

Terms of use:
openAccess
This article is made available under terms and conditions as specified in the corresponding bibliographic description in the repository

Publisher copyright

(Article begins on next page)
ON THE P vs NP QUESTION:
A PROOF OF INEQUALITY

Summary

The analysis discussed in this paper is based on a well-known NP-complete problem which is called “satisfiability problem or SAT”. From SAT a new NP-complete problem is derived, which is described by a Boolean function called “core function”. In this paper it is proved that the cost of the minimal implementation of core function increases with $n$ exponentially. Since the synthesis of core function is an NP-complete problem, this result is equivalent to proving that \( P \) and \( NP \) do not coincide.

1. INTRODUCTION

A brief description of the definitions and properties well known among the scientists of modern computational complexity theory which will be made reference to, is presented in this section.

\( P \) denotes the class of all the decision problems which can be solved in polynomial time.

\( NP \) denotes the class of all the decision problems \( f \) satisfying the property that the function \( \text{check}(f) \) analyzing a witness of the decision problem is polynomial time decidable.

“\( P=NP? \)”, or, in other terms, “Is \( P \) a proper subset of \( NP \)?”, is one of the most important open questions in modern computational complexity theory.

A decision problem \( C \) in \( NP \) is \( NP\text{-complete} \) if it is in \( NP \) and if every other problem \( L \) in \( NP \) is reducible to it, in the sense that there is a polynomial time algorithm which transforms instances of \( L \) into instances of \( C \) producing the same values.

The importance of NP-completeness derives from the fact that, if we find a polynomial time algorithm for just one \( NP\text{-complete} \) problem, then we can construct polynomial time algorithms for all the problems in \( NP \) and, conversely, if any single \( NP\text{-complete} \) problem does not have a polynomial time algorithm, than no \( NP\text{-complete} \) problem has a polynomial time solution.

The analysis discussed in this paper will be based on a well-known \( NP\text{-complete} \) problem which is called “satisfiability problem or SAT”.

Given a Boolean expression containing only the names of a set of variables (some of which may be complemented), the operators \( \text{AND}, \text{OR} \) and \( \text{NOT} \), and parentheses, is there an assignment of \( \text{TRUE} \) and \( \text{FALSE} \) values to the variables which makes the entire expression \( \text{TRUE} \)?
It is well known that the problem remains **NP-complete** also when all the expressions are written in "conjunctive normal form" with 3 variables per clause (problem **3SAT**). In this case, the analyzed expressions will be of the type:

$$F = (x_{11} \lor x_{12} \lor x_{13}) \land (x_{21} \lor x_{22} \lor x_{23}) \land \ldots \land (x_{t1} \lor x_{t2} \lor x_{t3})$$

(1)

where:
- \(t\) is the number of clauses or triplets;
- each \(x_{ij}\) is a variable in complemented or uncomplemented form;
- each variable can appear multiple times in the expression.

If the deterministic Turing machine is assumed as the computational model, with \(\{0,1,b\}\) as its set of input symbols, the input data appearing on the tape at the beginning of computation can represent the data of expression (1) in the following way:

$$b \ b \ b <\text{binary code of number of variables}> <\text{separator}> b$$

or

$$s_{11} \ n_{111} \ n_{112} \ n_{113} \ldots n_{11m}b$$
$$s_{12} \ n_{121} \ n_{122} \ n_{123} \ldots n_{12m}b$$
$$s_{13} \ n_{131} \ n_{132} \ n_{133} \ldots n_{13m}b$$
$$s_{21} \ n_{211} \ n_{212} \ n_{113} \ldots n_{21m}b$$

(2)

...  

$$s_{t3} \ n_{t31} \ n_{t32} \ n_{t33} \ldots n_{t3m}b$$

where:
- \(b\) is the blank symbol;
- \(t\) is the number of triplets;
- \(s_{ij}\) denotes the sign of variable \(x_{ij}\)
  (with \(s_{ij} = 1\) denoting that \(x_{ij}\) is preceded by operator **NOT**);
- \(n_{ijk}\) denotes the \(k\)-th component of the binary code \(<n_{ij1} \ n_{ij2} \ldots n_{ijm}>\) representing the name of variable \(x_{ij}\);
- the binary code of the number \(n_v\) of variables is needed in order to determine the number \(m\) of binary digits necessary to represent the names of variables according the rule \(m = \text{minimum integer not less than } \log_2 n_v\).

Notice that, by neglecting the bits of the binary code of the number of variables and the
bits of the separator, the number of input bits on the tape will be

\[ t \cdot 3 \cdot (1 + \text{minimum integer not smaller than } \log_2 (3 \cdot t)) \tag{3} \]

since the maximum value of the number of variables is \( 3 \cdot t \).

The properties of Turing machines processing the bit string described by (2) will be analyzed in this paper with reference to a family \( \{C_n\} \) of Boolean circuits, where \( C_n \) has \( n \) binary inputs and produces the same binary output as the corresponding Turing machine.

The equivalence between a deterministic Turing machine \( M \) processing some input \( x \) belonging to \( \{0,1\}^n \) and an \( n \)-input Boolean circuit \( C_n \) is well known. It is also known that the number of gates, or \textbf{AND, OR, NOT} operators appearing in circuit \( C_n \), is polynomial in the running time of the corresponding Turing machine.

2. **THE CORE FUNCTION**

In the case of satisfiability problem with 3 variables for clause, Boolean circuit \( C_n \) has \( n \) (=t) sets of inputs which the binary data described in (2) are applied to. (Of course, the binary code of the number of variables and the separator are not needed). The output of \( C_n \) (with \( n=t \)) will take the value \textbf{TRUE} when, and only when, there is an assignment of values \textbf{TRUE} and \textbf{FALSE} to variables making expression (1) \textbf{TRUE}.

In order to simplify analysis, circuit \( C_n \) will be decomposed into two processing layers as shown in Fig. 1, where, as usual, the number \( t \) of triplets plays the role of symbol \( n \) in the standard analysis of complexity theory.

In the following analysis, we shall use the symbol \( t \) when it’s necessary to remember the number of triplets and \( n \) in the other cases.

![Diagram of Boolean circuit decomposition](image.png)

Fig. 1

Decomposition of Boolean circuit \( C_n \) into compatibility layer and core layer
A variable \( j \) of triplet \( i \) will be defined as “compatible” with variable \( k \) of triplet \( h \) when, and only when either

- the sign \( s_{ij} \) of the former variable is equal to the sign \( s_{hk} \) of the latter,

or

- the name \(<n_{ij1} n_{ij2} ... n_{ijm}>\) of the former is different from the name \(<n_{hk1} n_{hk2} ... n_{hkm}>\) of the latter.

From that definition it follows that two “not compatible” variables have different signs and the same name; therefore, their AND are identically FALSE.

The compatibility layer is composed of \( 3 \cdot t \cdot (3 \cdot t - 3)/2 \) identical cells, one for each pair of variables belonging to different triplets.

As shown in Fig. 2, the inputs of a cell will be the sign \( s_{ij} \) and the name \(<n_{ij1} n_{ij2} ... n_{ijm}>\) of variable \( j \) of triplet \( i \), and the sign \( s_{hk} \) and the name \(<n_{hk1} n_{hk2} ... n_{hkm}>\) of variable \( k \) of triplet \( h \). The output of the same cell \( c(i,j;h,k) \) will be TRUE when, and only when, the two variables are compatible between themselves.

\[
\begin{align*}
\text{TRUE} & \iff x_{ij} \text{ is compatible with } x_{hk} \\
\text{FALSE} & \iff x_{ij} \text{ is not compatible with } x_{hk}
\end{align*}
\]

Variable \( c(i,j;h,k) \) will be called a compatibility variable or simply a compatibility.

The core layer processes only the \( 9 \cdot t \cdot (t-1)/2 \) compatibility variables \( c(i,j;h,k) \) and produces the global result of computation.

As the circuit \( C_n \), also the global Boolean function implemented by \( C_n \) may be decomposed into two layers of functions. At the compatibility layer, the function implemented by a cell may be written as follows (by using the symbols \( * \), \( + \), and \( ! \) for representing AND, OR and NOT operators, respectively):
\[ c(i,j;h,k) = s_{ij} \cdot s_{hk} + !s_{ij} \cdot !s_{hk} + n_{ij1} \cdot !n_{hk1} + !n_{ij1} \cdot n_{hk1} + n_{ij2} \cdot !n_{hk2} + !n_{ij2} \cdot n_{hk2} + \ldots + n_{ijm} \cdot !n_{hk1} + !n_{ijm} \cdot n_{hk1} \]

(4)

The Boolean function implemented by the core layer will be called the “Core Function” of order \( t \), where \( t \) is the number of triplets. It will be denoted with the symbol \( \text{CF}(t) \) or \( \text{CF}(n) \). The core function can be determined by proceeding as follows.

Consider one selection of variables appearing in (1), one and only one for each triplet, for all the triplets. Let

\[ <i_1>, <i_2>, \ldots, <i_t> \]

(5)

with \( i_1, i_2, \ldots, i_t \in \{1, 2, 3\} \)

be the indexes \textbf{<number of triplet, number of variable in the triplet>} of the selected variables. They will be called “characteristic indexes”. Let \( \Pi_k \) be the product of all the compatibility variables relative to the \textbf{k-th} of selections (5):

\[ \Pi_k = c(1,i_1; 2,i_2) \cdot c(1,i_1; 3,i_3) \cdot \ldots \cdot c(t-1,i_{t-1}; t, i_t) \]

(6)

The core function can be defined as the sum

\[ \Sigma_k \Pi_k \]

(7)

of the products (6) relative to all the selections (5).

For example, in the case of \( \text{CF}(3) \), the core function can be defined as follows:

\[ \text{CF}(3) = c(1,1;2,1) \cdot c(1,1;3,1) \cdot c(2,1;3,1) + c(1,1;2,1) \cdot c(1,1;3,2) \cdot c(2,1;3,2) + c(1,1;2,1) \cdot c(1,1;3,3) \cdot c(2,1;3,3) + c(1,1;2,2) \cdot c(1,1;3,1) \cdot c(2,2;3,1) + \ldots \]

(8)

...(other 22 products)... +

\[ c(1,3;2,3) \cdot c(1,3;3,3) \cdot c(2,3;3,3) \]

It is easy to prove that there is an assignment of value TRUE or FALSE to variables appearing in Eq. (1) which make the value of (1) equal to TRUE when, and only when, the core function takes the value TRUE.

Notice that the processing work of the cell of Fig. 2 increases as a polynomial function \( P(t) \) of the number of the variables since the increment of the length of the code of the name is logarithmic. Therefore, the total processing work of the compatibility layer increases as:

\[ 9 \cdot t \cdot (t - 1) \cdot P(t) \]
where $9 \cdot t \cdot (t - 1)/2$ is the total number of the compatibility cells.

Besides, the problem solved by the core layer is clearly in $\text{NP}$, because it is easy to verify a witness solution. It follows that, since the compatibility layer polynomially reduces an NP-complete problem ($\text{3SAT}$) to the problem solved by the core layer, the core function describes a new NP-complete problem.

Some interesting properties of core function have been discussed in ref. (23).

3. **A THEOREM OF BOOLEAN MONOTONIC FUNCTIONS**

Let $f(x_1, x_2, \ldots, x_t)$ be an isotonic Boolean function, that is a Boolean function which can be implemented with only AND and OR gates, applied to uncomplemented literals $x_1, x_2, \ldots, x_t$. It was believed that the minimum cost implementation of $f(x_1, x_2, \ldots, x_t)$ always contains only OR and AND gates, but A.Razborov proved that there are isotonic functions whose minimum cost implementation contains also NOT gates (see ref. (8)).

However, there is an upper bound on the comparison of the costs of the minimum cost implementations with and without NOT gates. It is specified by the following theorem.

3.1. **THEOREM**

Let $I_{\text{min}}$ be one of the minimum cost implementations of the isotonic Boolean function $f(x_1, x_2, \ldots, x_t)$, the cost being defined as the total number of AND, OR or NOT gates. Let $C_{\text{min}}$ be the cost of $I_{\text{min}}$.

There exists always an implementation $J$ of $f$ containing only AND and OR gates such that

$$\text{cost} (J) \leq 2 \cdot C_{\text{min}} + t$$

In order to prove this theorem, let us divide the gates of implementation $I_{\text{min}}$ of $f$ into different levels.

At level 1 we place the gates all inputs of which coincide with the complemented or uncomplemented input variables $x_i$ or $\overline{x_i}$ (where $\overline{x_i}$ denotes the complement of variable $x_i$).

Level 2 contains the gates whose inputs coincide with input variables or outputs of level 1 gates.

In general terms, level $q$ contains the gates whose inputs coincide with input variables or outputs of levels less than $q$.

We can transform $I_{\text{min}}$ into $J$ by deleting NOT gates and adding new AND or OR gates as follows.

We start from level 1.

For any level 1 AND gate we add an OR gate whose inputs are the complements of the inputs of the considered AND gate (Fig. 3). Similarly, for any level 1 OR gate we add an AND gate whose inputs are the complements of the corresponding OR gate.

By virtue of such operations, for any output $u$ of the level 1 gates a new node will be available in the new circuit we are generating whose value will be $\overline{u}$. 
As a second step of processing, for any level 2 AND gate of implementation $I_{\text{min}}$ we shall add an OR gate whose inputs are the complements of the inputs of the corresponding AND gate, in both the cases in which these inputs coincide with input variables of $f$ or with outputs of level 1 gates (Fig. 4).

A similar transformation will be applied to all level 2 OR gates.

As an example, the two level subnetwork of Fig. 5 will be transformed into the subnetwork of Fig. 6. Notice that at the outputs of $J$ not only the outputs $v$ and $w$ of $I_{\text{min}}$ will be available, but also their complements $\overline{v}$ and $\overline{w}$.

The preceding operations will be applied to all the levels of implementation $I_{\text{min}}$ in the order of increasing levels. It is apparent that, if for any input variable $x_i$ also $\overline{x_i}$ is available, the number of gates of $J$ is less than twice the number of gates of $I_{\text{min}}$. 

Fig. 3
The transformation of gates of level 1

Fig. 4
The transformation of gates of level 2
At level 0, before the gates of Fig. 6, t NOT gates might be necessary to generate the complemented input variables !x_i. Therefore, t has been added in the statement of the theorem.

This theorem will be very important in order to simplify the analysis of core function circuits.

4. PROPERTIES OF CORE FUNCTION

It is easy to prove the following properties of core function.

4.1. PROPERTY 1

Core function is totally isotone.

4.2. PROPERTY 2

Any product (6) is a prime implicant of core function (that is, a product of compatibilities ("PoC") which implies core function and no other term of it).

4.3. PROPERTY 3

Since the different selections of each of variables (5) are 3, the number of prime implicants of the core function is equal to 3^t. Each of these prime implicants is essential (that is, it does not imply a sum of other prime implicants) and it is the product of t(t-1)/2 compatibilities.
5. **PRODUCTS OF COMPATIBILITIES**

In the next section, reference will be made to the following definitions.

5.1. **DEFINITION OF SPURIOUS COMPATIBILITIES PAIR**

A pair of compatibility variables \( \{c(h,k;l,m), c(p,q;r,s)\} \) is defined as a spurious pair if

\[
\begin{align*}
& (h = p \text{ and } k \neq q) \\
\text{or} & (h = r \text{ and } k \neq s) \\
\text{or} & (l = p \text{ and } m \neq q) \\
\text{or} & (l = r \text{ and } m \neq s)
\end{align*}
\]

In a graphic scheme:

\[
\begin{array}{cccc}
\text{c( h, k ; l, m )} & \text{c( h, k ; l, m )} & \text{c( h, k ; l, m )} & \text{c( h, k ; l, m )} \\
\neq & \neq & \neq & \neq \\
\text{or} & \text{or} & \text{or} & \text{or} \\
\text{c( p, q ; r, s )} & \text{c( p, q ; r, s )} & \text{c( p, q ; r, s )} & \text{c( p, q ; r, s )}
\end{array}
\]

For example, the pair \( \{c(1,1;2,1), c(1,2;3,1)\} \) is a spurious pair since the triplet 1 is associated to two different indexes of variables (1 and 2).

5.2. **DEFINITION OF SPURIOUS PRODUCTS OF COMPATIBILITIES**

A spurious product of compatibilities (spurious PoC) is a product of compatibility variables containing the elements of one or more than one spurious pair.

For example, the PoC

\[
c(1,1;2,1) \ast c(1,2;3,1) \ast c(2,1;3,1)
\]

is a spurious PoC since it contains the elements of the spurious pair

\( \{c(1,1;2,1), c(1,2;3,1)\} \)

5.3. **DEFINITION OF IMPURE PRODUCTS OF COMPATIBILITIES**

A PoC containing one or more complemented variables will be defined as an impure PoC. In particular a term \( T \) of CF (that is, a PoC implying CF) that contains one or more complemented variables, will be defined as an impure term.

5.4. **DEFINITION OF CORE OF A PoC**

The product of all the uncomplemented variables of \( T \) will be defined as the core of \( T \).

5.5. **DEFINITION OF MARK**

Consider a not spurious subset of compatibilities satisfying the property that each of the indexes of triplet appears at least once in some variable. The product of the variables of such a
subset will defined as a “mark” of the prime implicant of which it contains a subset of compatibilities.

For example, in the case of CF (4), the PoC

\[ M = c(1,a;2,b) \cdot c(1,a;3,c) \cdot c(1,a;4,d) \]  

(9)

(\text{where } a, b, c, d \text{ are elements of } \{1,2,3\})

is a mark of the prime implicant

\[ P = c(1,a;2,b) \cdot c(1,a;3,c) \cdot c(1,a;4,d) \cdot c(2,b;3,c) \cdot c(2,b;4,d) \cdot c(3,c;4,d) \]  

(10)

since all the indexes of triplet appear at least once in (9).

5.6. \textbf{DEFINITION OF SPURIOUS MARK}

A spurious PoC in which all the indexes of triplet appear at least once will be called a “spurious mark”. Notice that a spurious mark may be the mark of more than one prime implicant. For the example, in the case of CF(3),

\[ c(1,1;2,1) \cdot c(1,1;3,1) \cdot c(1,1;2,2) \]

is a spurious mark of both the prime implicants

\[ c(1,1;2,1) \cdot c(1,1;3,1) \cdot c(2,1;3,1) \]

and

\[ c(1,1;2,2) \cdot c(1,1;3,1) \cdot c(2,2;3,1) \]

An impure PoC whose core is a (possibly spurious) mark will be a defined as a (possibly spurious) impure mark.

5.7. \textbf{DEFINITION OF EXTENDED PRIME IMPLICANTS}

A term T of core function, that is, an implicant of core function (a product of literals implying core function), contains all the uncomplemented literals of a prime implicant. Therefore, it may be defined as an “extended prime implicant” (only) to remember that it contains all the compatibilities of a prime implicant.

It may be a spurious extended prime implicant or an impure extended prime implicant or both a spurious and impure extended prime implicant.

Notice that an extended prime implicant can be viewed as a (possibly spurious or impure) mark.

5.8. \textbf{DEFINITION OF REMAINDER}

A PoC which is neither a (possibly spurious or impure) mark nor an (extended) prime implicant will be called a “remainder”. A remainder can be associated to one or more prime implicants, of which it contains a subset of compatibilities.

For example, in the case of CF(4)

\[ R = c(2,b;3,c) \cdot c(2,b;4,d) \cdot c(3,c;4,d) \]  

(11)

is a remainder of the prime implicant (10).
A remainder $R$ may be associated to more than one prime implicant. For example, in the case of $\text{CF}(3)$, $R = c(2,1;3,1)$ is a remainder of the prime implicants

\[
P_1 = c(1,1;2,1) \ast c(1,1;3,1) \ast c(2,1;3,1)
\]
\[
P_2 = c(1,2;2,1) \ast c(1,2;3,1) \ast c(2,1;3,1)
\]
\[
P_3 = c(1,3;2,1) \ast c(1,3;3,1) \ast c(2,1;3,1)
\]

(12)

On the definitions of mark and remainder the following properties are based.

5.9. **Property 4**

A not spurious mark $M$ specifies a corresponding prime implicant $P$ uniquely. Indeed, if all the indexes of triplet appear in $M$, the product (6) is completely defined.

We shall write

\[P = I(M)\]

to state that $P$ is the prime implicant specified by $M$.

As already mentioned, a remainder $R$ does not specify a corresponding prime implicant uniquely. In the example relative to $\text{CF}(3)$ above described, three prime implicants correspond to $R = c(2,1;3,1)$, as shown by (12), since a single index of triplet is missing in that remainder. In general, if $z$ triplets are not involved in $R$, there are $3^z$ different ways of involving the missing triplets.

Hence the following property follows.

5.10. **Property 5**

A not spurious remainder $R$ in which the indexes of $z$ triplets are missing corresponds to $3^z$ different prime implicants.

Finally, the following property can be proved. The proof is not too difficult and it is omitted for the sake of brevity.

5.11. **Property 6**

Let $P_1$ and $P_2$ be two PoC’s such that $P_1 \ast P_2$ is equal to a prime implicant $P$ of a core function. Either $P_1$ or $P_2$ is a mark of $P$. 
6. THE EXTERNAL CORE FUNCTION

Let \( I_j \) be a prime implicant of \( CF(n) \). The external core function relative to \( I_j \), \( ECF(n, I_j) \), is defined as the sum of all the minterms of \( CF(n) \) which imply \( I_j \) and no other prime implicant \( I_k \) of \( CF(n) \) with \( k \neq j \). (Remember that a minterm of a Boolean function \( F \) is a product of all the variables of \( F \), some complemented and some uncomplemented, implying \( F \)).

Of course,

\[
ECF(n, I_j) = I_j \cdot \prod_{k \neq j} (\neg I_k)
\]

where \( \neg I_k \) denotes the complement of \( I_k \), i.e. \( \text{NOT } I_k \).

The global external core function of order \( n \), or \( ECF(n) \), will be defined as the sum of \( ECF(n, I_j) \)'s relative to all the prime implicants \( I_j \) of \( CF(n) \):

\[
ECF(n) = \sum_j ECF(n, I_j)
\]

The importance of external core function derives from the following analysis.

6.1. THEOREM 1

Let \( T \) be a term (or extended prime implicant) of \( CF(n) \). It must be the product of all the compatibilities of a prime implicant \( I_j \) of \( CF(n) \) and other compatibilities, that is,

\[
T = I_j \cdot X
\]

where \( X \) is a possibly empty PoC, which can also be written as \( T = T(I_j) \)

All the minterms of \( T(I_j) \) contained in \( ECF(n) \) are minterms of \( ECF(n, I_j) \).

Indeed, for any \( k \neq j \),

\[
T(I_j) \cdot ECF(n, I_k) = I_j \cdot X \cdot I_k \cdot \prod_{l \neq k} (\neg I_l) = 0
\]

6.2. THEOREM 2

Let \( T \) be a term of \( CF(n) \) implying two or more prime implicants of \( CF(n) \) as, for example,

\[
T = T(I_j, I_k)
\]

The number of minterms of \( T(I_j, I_k) \) belonging to \( ECF(n) \) is equal to 0.

Indeed,

\[
T(I_j, I_k) \cdot ECF(n, I_h) = 0
\]

for any \( h \).

The preceding theorems 1 and 2 are nearly obvious. On the contrary, the following theorem 3 appears rather complex.
6.3. **Theorem 3**

Let $T = T (Ij) = Ij * X$ be a term of $CF (n)$ which is spurious for a single compatibility $X$.

If $NMT(F)$ denotes the number of minterms of Boolean function $F$, the number of minterms of $Ij * X$ contained in $ECF(n, Ij)$ is

$$NMT(Ij * X * ECF(n, Ij)) \leq \frac{1}{2} \cdot NMT(ECF(n, Ij))$$

(17)

**Proof**

The number of minterms of $T$ contained in $ECF(n, Ij)$ is equal to the number of minterms contained in

$$Ij * X * ECF(n, Ij) = Ij * X * \Pi_{k \neq j} (I_k) = Ij * X * (A * (!X) + B) = Ij * X * B$$

(18)

where $A$ and $B$ are two antitone functions (that is, two monotone functions which can be described with complemented variables only) containing neither $X$ nor $!X$.

The number of minterms of $ECF(n, Ij)$ is equal to the number of minterms contained in

$$Ij * \Pi_{k \neq j} (I_k) = Ij * A * (!X) + Ij * B$$

(19)

Besides,

$$NMT(Ij * B) = 2 \cdot NMT(Ij * X * B)$$

(20)

since $X$ appears neither in $B$ nor in $Ij$.

The statement of this Theorem 3 derives from the comparison of (18), (19) and (20).

By proceeding in the same way it is possible to generalize the preceding Theorem 3 as follows.

6.4. **Theorem 4**

Let

$Ij * X_1 * X_2 * ... * X_m$

are $m$ spurious compatibilities.

The number of its minterms contained in $ECF(n, Ij)$ is

$$NMT(Ij * X_1 * X_2 * ... * X_m * ECF(n, Ij)) \leq \frac{1}{2^m} \cdot NMT(ECF(n, Ij))$$

(21)

**Proof**

For the sake of brevity, the proof of (21) is restricted to the case $m = 2$.

In this case we can write:

$$ECF(n, Ij) = Ij * A * (!X_1) + Ij * B * (!X_2) + Ij * C * (!X_1) * (!X_2) + Ij * D$$

(22)

where functions $A$, $B$, $C$, $D$ do not contain variables $X_1$ or $X_2$.

Notice that $X_1 * X_2 * ECF(n, Ij) = X_1 * X_2 * Ij * D$
and \((X_1 \cdot X_2 \cdot D)\) contains \(\frac{1}{4}\) of the minterms of \(D\).

From these two remarks the statement of Theorem 4 derives.

The following Theorems 5 and 6 are analogous to preceding Theorems 3 and 4, respectively.

### 6.5. **Theorem 5**

Let \(T = T(I_j)\) an impure term of \(\text{CF}(n)\) characterized by a single impure variable \(!X\):

\[ T = I_j \cdot (!X) \]

The number of minterms of \(\text{ECF}(n, I_j)\) contained in \(T\) is

\[
\text{NMT} \left( I_j \cdot (!X) \cdot \text{ECF}(n, I_j) \right) \leq \left( \frac{1}{2} + \frac{K(n)}{2} \right) \cdot \text{NMT} \left( \text{ECF}(n, I_j) \right)
\]  

(23)

where \(K(n)\) is positive and less than 1 and it is a quickly decreasing function of \(n\). The proof of **Theorem 5** and the properties of function \(K(n)\) are discussed in Appendix 1.

### 6.6. **Theorem 6**

Let \(T = T(I_j)\) an impure term of \(\text{CF}(n)\) characterized by \(m\) impure variables:

\[ T = I_j \cdot (!X_1) \cdot (!X_2) \cdot \ldots \cdot (!X_m) \]

The number of minterms of \(\text{ECF}(n, I_j)\) contained in \(T\) is

\[
\text{NMT} \left( T \cdot \text{ECF}(n, I_j) \right) \leq \left( \frac{1}{2} + \frac{K(n)}{2} \right)^m \cdot \text{NMT} \left( \text{ECF}(n, I_j) \right)
\]

(24)

Also Theorem 6 is discussed in Appendix 1.

Notice that \(\text{NMT}(\text{ECF}(n, I_j)) = \text{NMT}(\text{ECF}(n, I_k))\) for any \(j\) and \(k\). It will be called **NMT1**.
7. THE REFERENCE ARCHITECTURE

Fig. 7 shows the network which will implement core function. It is characterized by a number of subnetworks each of which has the structure shown by Fig. 8. As an alternative, the network of Fig. 7 might be composed by a single network of the type of Fig. 8.

![Diagram of the Reference Architecture]

Fig. 7
The Reference Architecture
The circuit presented in Fig. 8 will be called a “primary composite addendum (PCA)”. Every $F_i$ will be called a “primary composite addendum factor” (PCAF).

![Diagram of PCA]

The primary composite addendum

If the number of PCA's of the minimum cost implementation of $CF(n)$ increased with $n$ according to an exponential law, also the cost of this implementation would increase according to an exponential law, the cost being represented by the number of AND gates at the bottom of Fig. 7.

Therefore, the following analysis refers to the case in which the number of PCA's of the minimum cost implementation of $CF(n)$ increases with $n$ according to a polynomial law.

Besides, reference will be made to the following definitions. The merit of a (possibly, impure or spurious) prime implicant $P_i$ of $CF(n)$ will be defined as the number of minterms of $ECF(n)$ that $P_i$ covers and the merit of a PCA will be defined as the number of minterms of $ECF(n)$ that this PCA covers.

We shall discuss the properties of the PCA which contains the maximum number of minterms of $ECF(n)$. It will be called $PCAMAX$.

It is easy to prove that the number of minterms of $ECF(n)$ contained in the function implemented by $PCAMAX$ increases with $n$ as $3^n$. Besides, also the number of prime implicants of $CF(n)$ implemented by $PCAMAX$ increases with $n$ as $3^n$. 
8. SYNTHESIS OF MAXIMUM MERIT PCA

Consider the decomposition of the maximum merit PCA into \( k \) factors
\[ F_1, F_2, F_3, ..., F_k \]
(Fig. 8). Consider also the partial product
\[ F_{2-k} = F_2 \ast F_3 \ast \ldots \ast F_k \]

where the symbols \( F_2, F_3, ..., F_k \), above used to denote processing units have the meaning of their corresponding Boolean values, as will be done in the future when such a choice will not generate confusion.

Obviously, the value of the maximum merit PCA, that is, the function implemented by it, will be
\[ \text{val}(\text{PCAMAX}) = F_1 \ast F_{2-k} \]

Let \( P_1, P_2, ..., P_v \) be the prime implicants of function \( F_1 \) and \( Q_1, Q_2, ..., Q_w \) the prime implicants of function \( F_{2-k} \). Obviously, the value of the maximum merit PCA will be the sum of all the \( v \ast w \) products \( P_i \ast Q_j \). Some of these products will be equal to 0; the other ones will be (possibly, impure or spurious) implicants of \( \text{CF}(n) \).

The number of minterms of \( \text{ECF}(n) \) covered by each of these implicants will be defined as its merit.

Notice that any product \( P_i \ast Q_j \) “must” be an implicant of \( \text{CF}(n) \) (possibly, extended with spurious or impure variables). Otherwise, the considered solution would not be a correct implementation of \( \text{CF}(n) \).

Fig. 9 shows the symbols which will be used in the following analysis.

An arc connecting node \( P_i \) with node \( Q_j \) denotes that the product \( P_i \ast Q_j \) is a (possibly impure or spurious) implicant of \( \text{CF}(t) \). For example, this is the case of arcs \( P_1 \ast Q_1, P_1 \ast Q_2, P_2 \ast Q_1, P_2 \ast Q_2 \) in Fig. 9. The labels of the arcs \( I_0, I_1, I_2, I_3, I_0' \) (perhaps, the same as \( I_0 \)), \( I_1' \) are the names of the prime implicants represented by those arcs. A missing arc denotes that the corresponding product is equal to 0; thus, for example, \( P_1 \ast Q_3 = 0 \) or \( P_4 \ast Q_2 = 0 \).

Notice that an arc might be labelled with the product of two or more different prime implicants, as in the case of \( P_4 \ast Q_4 \) which has been labelled with the product \( I_3 \ast I_4 \). However, as already proved, the merit of the product of two or more prime implicants is equal to 0.
Three different cases are worth mentioning.

**Case 1.**

Both $P_3$ and $Q_3$ are marks, and $I(P_3) = I(Q_3)$. Of course, in this case, $I_2 = I(P_3) = I(Q_3)$. Notice that, by virtue of Property 6 of previous Section 5, if $P_3 \ast Q_3$ is not equal to 0, at least one of these two terms is a mark of the generated prime implicant.

**Case 2**

$P_2$ is a mark and $Q_2$ is a remainder. Obviously, $I_1 = I(P_2)$. The considered arc is oriented from $P_2$ to $Q_2$ in order to remember that $P_2$ is the “origin” of the arc, that is, the mark of the corresponding prime implicant.

This is also the case of the arcs $P_1 - Q_1$, $P_1 - Q_2$, $P_2 - Q_1$.

Notice that in **Case 1** both $P_3$ and $Q_3$ might be considered as origins of the prime implicant $I_2 = P_3 \ast Q_3$.

**Case 3**

$P_5$ is a mark of a prime implicant $I(P_5)$ while $Q_5$ is a mark of a different prime implicant $I(Q_5) \neq I(P_5)$. Since the produced prime implicant $I_5$ coincides with $I(Q_5)$, the arc has been oriented from $Q_5$ which is considered as the origin of the arc.

Since the number of prime implicants implemented by $\text{PCA}_\text{MAX}$ increases with $n$ as $3^n$, also the number of origins born in the decomposition of Fig. 5 increases with $n$ as $3^n$. 
Assume that the number of origins labeled as $Q_j$ is larger than the number of origins labeled as $P_i$. In this case, of course, the number of $Q_j$ origins increases with $n$ as $3^n$. The case in which the number of $Q_j$ origins is less than the number of $P_i$ origins can be treated in a similar way.

We can organize the $Q_j$ origins as follows.

Let us start from an origin $Q_{11}$ and a node $P_1$ such that $Q_{11} \ast P_1 \neq 0$.

Collect all the origins $Q_{12}, Q_{13}, \ldots$ such that $Q_{1j} \ast P_1 \neq 0$. Of course, for all the remaining $Q_{ij}$, $Q_{ij} \ast P_1 = 0$.

Now consider a new $P_2$ and a new $Q_{21}$ such that $Q_{21} \ast P_2 \neq 0$. Collect all the origins $Q_{22}, Q_{23}, \ldots$ such that $Q_{2j} \ast P_2 \neq 0$. Repeat the same procedure till all $P_i$'s and all $Q_{ij}$'s have been involved.

In Appendix 2 it is shown that the number of such subsystems cannot increase with $n$ exponentially, if we accept the hypothesis that the number of minterms of $ECF(n)$ contained in $PCA_{MAX}$ increases with $n$ as $(3^n) \cdot NMT1(n)$. It follows that the prime implicants of at least one of the subsystems must cover a number of minterms of $ECF(n)$ increasing with $n$ as $(3^n) \cdot NMT1(n)$. We can define this subsystem as “the most effective subsystem”.

9. THE SYNTHESIS OF THE MOST EFFECTIVE SUBSYSTEM

Let us assume that $<P_1, \{Q_{ij}\}>$ is the most effective subsystem.

Assume that $Q_{11}$ is complete in all the compatibilities which characterize it with respect to the other $Q_{ij}$'s.

For example, if index $<1,1>$ (input 1 of triplet 1) appears in $Q_{11}$ and index $<1,2>$ (input 2 of triplet 1) appears in $Q_{12}$, then $Q_{11}$ contains all the compatibilities involving $<1,1>$. Similarly, each of the other $Q_{ij}$'s is complete in all the indexes which characterize it with respect to the other $Q_{ij}$'s.

Such subsystem will be defined as “complete”.

A simple example of a complete subsystem relative to $CF(3)$ is:

$$P_1 = c(2,1;3,1)$$
$$Q_{11} = c(1,1;2,1) \ast c(1,1;3,1)$$
$$Q_{12} = c(1,2;2,1) \ast c(1,2;3,1)$$
$$Q_{13} = c(1,3;2,1) \ast c(1,3;3,1)$$

(25)

If the considered subsystem is complete, its merit is exactly the sum of the numbers of minterms of $ECF(n)$ contained in all the products $P_1 \ast Q_{ij}$'s.

Since the number of prime implicants implemented by $PCA_{MAX}$ increases with $n$ as $3^n$, also the number of origins born in the decomposition of Fig. 9 increases with $n$ as $3^n$.

Let us assume that one origin $Q_{ij}$ is not complete in all the compatibilities involving an index which does not appear in other origins of the same subsystem. For example, in the case of subsystem (25) of $CF(3)$, assume that $Q_{11}$ does not contain $c(1,1;3,1)$. In that case, $P_1$ should contain $c(1,1;3,1)$ and the merits of $Q_{12}$ and $Q_{13}$ would be reduced to half of their original value (because of Theorem 3 of Section 6).
In more general terms consider the following complete system relative to \( \text{CF}(6) \):

\[
P_1 = c(1,1;2,1) * c(1,1;3,1) * c(2,1;3,1)
\]

\[
Q_{11} = c(1,1;4,1) * c(1,1;5,1) * c(1,1;6,1) *
\]

\[
c(2,1;4,1) * c(2,1;5,1) * c(2,1;6,1) *
\]

\[
c(3,1;4,1) * c(3,1;5,1) * c(3,1;6,1) *
\]

\[
c(4,1;5,1) * c(4,1;6,1) * c(5,1;6,1)
\]

\[
Q_{12} = \ldots
\]

\[
Q_{13} = \ldots
\]

\[
\ldots \ldots
\]

Assume that \( c(1,1;4,1) \) is cancelled in \( Q_{11} \). As a consequence, \( P_1 \) must be multiplied by \( c(1,1;4,1) \) in order that \( P_1 * Q_{11} \) is still an implicant of \( \text{CF}(6) \).

This modification implies a reduction of the merits of many \( P_1 * Q_{1k} \)'s according to the following rules:

1) the merits of 1/3 of them remain unchanged
2) the merits of 2/3 of them are multiplied by 1/2.

Therefore, the total merit of the considered subsystem will be reduced of \((1/3) \cdot 1 + (2/3) \cdot (1/2) = 2/3\)

Then assume that both the compatibilities \( c(1,1;4,1) \) and \( c(1,1;5,1) \) are cancelled in some of \( Q_{1k} \)'s and \( P_1 \) is multiplied by \( c(1,1;4,1) * c(1,1;5,1) \).

As a consequence of these modifications the merits of the \( P_1 * Q_{1k} \)'s will be changed as follows:

1) the merits of 1/9 of them will remain unchanged;
2) the merits of 4/9 of them will be multiplied by 1/2;
3) the merits of 4/9 of them will be multiplied by 1/4.

Therefore, the total merit of \( P_1 * \sum Q_{1k} \) will be multiplied by \( 4/9 = (2/3)^2 \).

In more general terms, it is easy to prove that if \( q \) compatibilities necessary to completeness are missing in \( Q_{1k} \) the total merit of the considered subsystem will be reduced to the extent of \( (2/3)^q \).

In general terms, a subsystem characterized by \( m \) fixed indexes in \( P_i \) may contain \( 3^{n-m} \) prime implicants (or less). Therefore, if this subsystem is complete, its total merit may reach the value

\[
M = 3^{n-m} \cdot \text{NMT1}(n)
\]

If \( q \) compatibility are missing, the merit becomes \( (2/3)^q \cdot M \)

Very few not complete \( Q_{ij} \)'s would reduce the merit of the considered subsystem to values not compatible with the hypothesis that the merit of the subsystem increases as \( 3^n \cdot \text{NMT1}(n) \). Neither \( q \) nor \( m \) can be increasing functions of \( n \).
10. THE SYNTHESIS OF A COMPLETE SUBSYSTEM

Assume that PCA_MAX is a complete subsystem of CF(n) covering a number of minterms of ECF(n) of the order of 3^n \cdot NMT1(n). In this case

\[ F_{2-k} = Q_1 + Q_2 + ... + Q_z \]

where \( z \) increase with \( n \) as \( 3^n \) and each \( Q_i \) is a mark and it is complete in all the indexes which characterize it.

Now consider the decomposition

\[ F_{2-k} = F_2 \cdot F_{3-k} \]

where \( F_2 \) and \( F_{3-k} \) can be written as sums of their prime implicants

\[ F_2 = R_1 + R_2 + ... \]

\[ F_{3-k} = S_1 + S_2 + ... \]

It is easy to prove that \( F_2 \) and \( F_{3-k} \) must contain at least \( (z-1) \) marks, that is, in the best case the decomposition of \( F_{2-k} \) produces the reduction by one unit of the number of marks contained in \( F_{2-k} \).

Indeed, assume, for example, that

\[ R_1 \cdot S_1 = Q_1 \]

\[ R_2 \cdot S_2 = Q_2 \]

where \( R_1, R_2, S_1 \) and \( S_2 \) are all remainders. It is easy to verify that if \( R_1, R_2, S_1 \) and \( S_2 \) are all remainders, \( R_1 \cdot S_2 \cdot P_1 \) and \( R_2 \cdot S_1 \cdot P_1 \) are not implicants of CF(n).

Indeed, for example, if CF(5) is the considered core function and

\[ P_i = c(1,1;2,1) \cdot c(1,1;3,1) \cdot c(2,1;3,1) \]

\[ Q_1 = c(1,1;4,1) \cdot c(1,1;5,1) \cdot c(2,1;4,1) \cdot c(2,1;5,1) \cdot c(3,1;4,1) \cdot c(3,1;5,1) \cdot c(4,1;5,1) \]

\[ Q_2 = c(1,1;4,2) \cdot c(1,1;5,1) \cdot c(2,1;4,2) \cdot c(2,1;5,1) \cdot c(3,1;4,2) \cdot c(3,1;5,1) \cdot c(4,2;5,1) \]

\[ R_1 = c(1,1;4,1) \cdot c(1,1;5,1) \cdot c(2,1;4,1) \cdot c(2,1;5,1) \cdot c(4,1;5,1) \]

\[ (which\ is\ a\ remainder\ since\ <3,1>\ is\ missing) \]

\[ S_1 = c(2,1;4,1) \cdot c(2,1;5,1) \cdot c(3,1;4,1) \cdot c(3,1;5,1) \cdot c(4,1;5,1) \]

\[ (which\ is\ a\ remainder\ since\ <1,1>\ is\ missing) \]

\[ R_2 = c(1,1;4,2) \cdot c(1,1;5,1) \cdot c(2,1;4,2) \cdot c(2,1;5,1) \cdot c(4,2;5,1) \]

\[ (which\ is\ a\ remainder\ since\ <3,1>\ is\ missing) \]

\[ S_2 = c(2,1;4,2) \cdot c(2,1;5,1) \cdot c(3,1;4,2) \cdot c(3,1;5,1) \cdot c(4,2;5,1) \]

\[ (which\ is\ a\ remainder\ since\ <1,1>\ is\ missing) \]

The product \( P_i \cdot R_1 \cdot S_2 \) does not imply \( P_i \cdot Q_1 \) since \( c(3,1;4,1) \) is missing and it does not imply \( P_i \cdot Q_2 \) since \( c(1,1;4,2) \) is missing, while the product \( P_i \cdot R_2 \cdot S_1 \) does not imply \( P_i \cdot Q_1 \) since \( c(1,1;4,1) \) is missing and it does not imply \( P_i \cdot Q_2 \) since \( c(3,1;4,2) \) is missing.
Similarly, the decomposition

\[ F_{3-k} = F_3 \ast F_{4-k} \]

in the best case produces the reduction by one unit of the number of marks contained in \( F_3 \).

It follows that the decomposition

\[ F_1 \ast F_2 \ast \ldots \ast F_k. \]

can generate the reduction of the number of marks by \( k-1 \), but it requires at least \( k \) gates, the OR gates of Fig. 8.

It is worth remarking that the preceding considerations hold also in the case of quasi-complete subsystems in which the absence of one or more of the few compatibilities in a \( Q_j \) determines the presence of the same compatibility in \( P_i \).

For example, in the case described by the set of equations (26), if \( P_1 \) were modified as follows:

\[ P'_1 = c(1,1;2,1) \ast (1,1;3,1) \ast c(2,1;3,1) \ast c(3,1;4,1) \ast c(3,1;4,2) \]

the double decomposition

\[ (R_1+R_2+G) \ast (S_1+S_2+G) \]

(where \( R_1, R_2, S_1 \) and \( S_2 \) are remainders) would be possible.

However, in this case, the merits of \( P'_1 \ast R_1 \ast S_1 \) and \( P'_1 \ast R_2 \ast S_2 \) (as well as the merits of many other implicants) are reduced to the half of \( NMT_1(n) \).

Also in this case one gate makes it possible to reduce the merit of the considered subsystem of one unit \( NMT_1(n) \).

The product \( F_1 \ast F_2 \ast \ldots \ast F_k \) may produce other marks in addition to those generated inherently by the product \( F_1 \ast F_2 \ast \ldots \ast F_k \). Indeed, one or more marks of \( CF(n) \) can be implicants of some \( F_j \).

For example, consider function \( F_1 \) implemented by the first of PCAF’s represented in Fig. 10.

\( F_1 \) is the output of an OR gate. Indeed, if it were the output of an AND gate, this might be merged together with the AND gate producing the output of the considered PCA with the reduction of the cost by one unit.

Let \( F_{11}, F_{12}, \ldots, F_{1l} \) be the inputs of this OR gate (Fig. 10). In its turn, node \( F_{11} \) contains a mark or a sum of marks as the product of functions \( F_{111}, F_{112}, F_{113}, \ldots \).
The considerations above developed on the AND gates producing $\text{PCA}_{\text{MAX}}$ (at the bottom of Fig. 10) apply also to the AND gate producing $F_{11}$. This gate can generate $h \cdot 1$ origins at the cost of $h$ gates, that is, the OR gates producing $F_{111}, F_{112}, F_{113}$...

11. CONCLUSION

In order to implement the sum of $T$ prime implicants of core function, the subnetwork $\text{PCA}_{\text{MAX}}$ must employ more than $T$ gates. Since the number of prime implicants implemented by $\text{PCA}_{\text{MAX}}$ increases with $n$ as $3^n$, the cost of the minimal implementation of core function $\text{CF}(n)$ increases with $n$ as $3^n$. Since the synthesis of core function is an NP-complete problem, this result is equivalent to proving that $P$ and NP do not coincide.
12. APPENDIX 1

12.1. PROOF OF THEOREM 5

If we consider all the prime implicants of \( \text{ECF}(n,I_1) \) and collect all the prime implicants which contain \((!X)\), where \( X \) is variable not contained in \( I_1 \), we can write:

\[
\text{ECF}(n,I_1) = I_1 + \Pi_{k \neq 1} (!I_k) = I_1 + ((!X)*F+G)
\]

where \( F \) and \( G \) are two antitone functions (that is, Boolean functions which can be written as sums of products of only complemented variables) not containing variable \( X \).

Let

\[
K(n) \cdot \text{NMT}(\text{ECF}(n,I_1))
\]

be the number of the minterms of \( \text{ECF}(n,I_1) \) contained in \((!X)*F\)

and

\[
(1-K(n)) \cdot \text{NMT}(\text{ECF}(n,I_1))
\]

be the number of the minterms of \( \text{ECF}(n,I_1) \) contained in \( G \) and not contained in \((!X)*F\).

Now consider an impure term

\[
T = T(I_1) = (!X)*I_1
\]

Since the number of minterms of \( T \) contained in \((!X)*G\) is the half of the minterms contained in \( G \), the number of minterms of \( T \) contained in \( \text{ECF}(n,I_1) \) is

\[
\text{NMT}( (!X)*I_1 + \text{ECF}(n,I_1) ) = \text{NMT}(I_1 + (!X)*F + I_1 + (!X)*G) < \\
< (K(n) + (1/2) \cdot (1-K(n)) \cdot \text{NMT}(\text{ECF}(n,I_1))) = \\
= (1/2 + K(n)/2) \cdot \text{NMT}(\text{ECF}(n,I_1))
\]

where sign < is due to the fact that \( F \) and \( G \) have common minterms.

12.2. PROOF OF THEOREM 6

Assume that the number of minterms of \( \text{ECF}(n,I_1) \) containing \((!X_1) \) (or \((!X_2) \) ) is equal to \( K(n) \cdot \text{NMT}(\text{ECF}(n,I_1)) \).

It is easy to verify that the number of minterms of \( \text{ECF}(n,I_1) \) containing both the variables \((!X_1) \) and \( X_2 \) is

\[
\text{NMT}(\text{ECF}(n,I_1) + (!X_1)*X_2) <= K(n) \cdot (1-K(n)) \cdot \text{NMT}(\text{ECF}(n,I_1))
\]

Similarly,

\[
\text{NMT}(\text{ECF}(n,I_1) + (!X_2)*X_1) <= (1-K(n)) \cdot K(n) \cdot \text{NMT}(\text{ECF}(n,I_1))
\]

\[
\text{NMT}(\text{ECF}(n,I_1) + X_1*X_2) <= (1-K(n)) \cdot (1-K(n)) \cdot \text{NMT}(\text{ECF}(n,I_1))
\]

\[
\text{NMT}(\text{ECF}(n,I_1) + (!X_2)*(!X_1)) <= K(n) \cdot K(n) \cdot \text{NMT}(\text{ECF}(n,I_1))
\]

It follows that the number of minterms of \( T(I_1) = (!X_1)*(!X_2)*I_1 \) contained in \( \text{ECF}(n,I_1) \) is less or equal to

\[
K(n) \cdot (1-K(n)) \cdot \text{NMT}(\text{ECF}(n,I_1))/2 + \\
(1-K(n)) \cdot K(n) \cdot \text{NMT}(\text{ECF}(n,I_1))/2 +
\]
\[(1-K(n)) \cdot (1-K(n)) \cdot \text{NMT(ECF(n,I_1))}/4 + \]
\[K(n) \cdot K(n) \cdot \text{NMT(ECF(n,I_1))}=\]
Therefore, the number of minterms of \(T(I_1) = (\neg X_1) \cdot (\neg X_2) \cdot I_1\) contained in \(\text{ECF(n,I_1)}\) is less or equal to
\[\left(\frac{1}{2} + \frac{K(n)}{2}\right)^2 \cdot \text{NMT(ECF(n,I_1))}\]
which is equivalent to eq. (21) for \(m=2\).
The extension to any value of \(m\) can be easily performed by applying the same technique.

12.3. **Evaluation of \(K(n)\)**

In order to evaluate \(K(n)\) consider the simple case of \(n=3\) with
\(I_1 = c(1,1;2,1) \cdot c(1,1;3,1) \cdot c(2,1;3,1)\).

In this case
\[\text{ECF}(3,I_1) = I_1 \cdot (\neg I_2) \cdot (\neg I_3) \cdot \ldots \cdot (\neg I_{27})\]
where
\[\neg I_2 = \neg c(1,1;2,1) + \neg c(1,1;3,2) + \neg c(2,1;3,2)\]
\[\neg I_3 = \neg c(1,1;2,1) + \neg c(1,1;3,3) + \neg c(2,1;3,3)\]
\[\neg I_4 = \neg c(1,1;2,2) + \neg c(1,1;3,1) + \neg c(2,2;3,1)\]
\[\neg I_5 = \neg c(1,1;2,2) + \neg c(1,1;3,2) + \neg c(2,2;3,2)\]
\[\neg I_6 = \neg c(1,1;2,2) + \neg c(1,1;3,3) + \neg c(2,2;3,3)\]
\[\neg I_7 = \neg c(1,1;2,3) + \neg c(1,1;3,1) + \neg c(2,3;3,1)\]
\[\neg I_8 = \neg c(1,1;2,3) + \neg c(1,1;3,2) + \neg c(2,3;3,2)\]
\[\neg I_9 = \neg c(1,1;2,3) + \neg c(1,1;3,3) + \neg c(2,3;3,3)\]
\[\neg I_{10} = \neg c(1,2;2,1) + \neg c(1,2;3,1) + \neg c(2,1;3,1)\]
\[\ldots\]
\[\neg I_{27} = \neg c(1,3;2,3) + \neg c(1,3;3,3) + \neg c(2,3;3,3)\]

Consider the two following functions:
\[H = \neg c(1,1;2,2) + \]
\[+ \neg c(2,2;3,1) \cdot \neg c(1,1;2,3) + \neg c(1,1;3,3) + \]
\[+ \neg c(2,2;3,1) \cdot \neg c(1,1;3,2) + \neg c(2,2;3,3) + \]
\[+ \neg c(2,2;3,1) \cdot \neg c(2,2;3,2) + \neg c(1,1;3,3) + \]
\[+ \neg c(2,2;3,1) \cdot \neg c(2,2;3,2) + \neg c(2,2;3,3)\]
\[F = I_1 \cdot (\neg I_2) \cdot (\neg I_3) \cdot (\neg I_7) \cdot (\neg I_8) \cdot \ldots \cdot (\neg I_{27})\]
from which the following equation derives:

\[ ECF(3,I1) = F \cdot !c(1,1;2,2) + F \cdot H \]

where \( F \) and \( H \) do not contain variable \( c(1,1;2,2) \).

Function \( ECF(3,I1) \) is the sum of the three following functions:

\[ F_1 = !c(1,1;2,2) \cdot F \cdot (!H) \]
\[ F_2 = !c(1,1;2,2) \cdot F \cdot H \]
\[ F_3 = c(1,1;2,2) \cdot F \cdot H \]

These functions are disjoint in the sense that \( F_i \) contains none of the minterms contained in \( F_j \) with \( j <> i \). Therefore,

\[ NMT(ECF(3,I1)) = NMT(F_1) + NMT(F_2) + NMT(F_3) \]

Since function \( F \) is antitonic in all the variables contained in \( H \), for any minterm of \( F_1 \) there is a minterm of \( F_2 \). Therefore,

\[ NMT(F_1) <= NMT(F_2) \]

Besides,

\[ NMT(F_2) = NMT(F_3) \]

Therefore,

\[ K(3) = (NMT(F_1) + NMT(F_2)) / NMT(ECF(3,I1)) <= (2/3) \]

In more general terms

\[ K(n) <= (2/3) \]

Consider a prime implicant \( P \) of \( F \) and assume that it contains \( q \) minterms of \( ECF(3,I1) \).

The product \( F \cdot !c(1,1;2,2) \) produces \( q/2 \) minterms of \( ECF(3,I1) \), since the minterms of \( F \) containing \( c(1,1;2,2) \) are not minterms of \( F \cdot !c(1,1;2,2) \).

The same prime implicant \( P \) of \( F \) produces a variable number of minterms after the multiplication by each of the five addenda of \( H \). For example, the multiplication

\[ P \cdot !c(2,2;3,1) \cdot !c(1,1;3,2) \cdot !c(1,1;3,3) \]

produces \( q/8 \) minterms of \( ECF(3,I1) \) if \( P \) contains none of the three variables

\[ !c(2,2;3,1) ; \quad !c(1,1;3,2) ; \quad !c(1,1;3,3) \]

but it produces \( q \) minterms of \( ECF(3,I1) \) if it contains all the three considered variables. Besides, the minterms generated by the product \( P \cdot !c(2,2;3,1) \cdot !c(1,1;3,2) \cdot !c(1,1;3,3) \) must be added to those generated by the other addenda of \( H \).

It is apparent that \( K(3) \) is larger than \( 0.5 \).

Besides, it is a function which decreases when \( n \) increases.

Indeed, when \( n \) increases, the number of addenda of \( H \) increases.
13. **APPENDIX 2**

In order to understand what will be proved in this Appendix assume, for the sake of simplicity, that there is a single **PCA** in the implementation of **CF(4)** and therefore **PCA_{MAX} = CF(4)**.

Assume also that the **PoC's P_i** take the following values:

\[ P_1 = c(1,1;2,1) \cdot c(1,1;3,1) \cdot c(2,1;3,1) \]
\[ P_2 = c(1,1;2,1) \cdot c(1,1;3,2) \cdot c(2,1;3,2) \]
\[ P_3 = c(1,1;2,1) \cdot c(1,1;3,3) \cdot c(2,1;3,3) \]
\[ P_4 = c(1,1;2,2) \cdot c(1,1;3,1) \cdot c(2,2;3,1) \]
\[
\ldots
\]
\[ P_{27} = c(1,3;2,3) \cdot c(1,3;3,3) \cdot c(2,3;3,3) \]

Let the origins associated to **P_1** be the following ones:

\[ Q_{11} = c(1,1;4,1) \cdot c(2,1;4,1) \cdot c(3,1;4,1) \]
\[ Q_{12} = c(1,1;4,2) \cdot c(2,1;4,2) \cdot c(3,1;4,2) \]
\[ Q_{13} = c(1,1;4,3) \cdot c(2,1;4,3) \cdot c(3,1;4,3) \]

Now consider the **Q_{2j}'s** associated to **P_2**:

\[ Q_{21} = c(1,1;4,1) \cdot c(2,1;4,1) \cdot c(3,2;4,1) \]
\[ Q_{22} = c(1,1;4,2) \cdot c(2,1;4,2) \cdot c(3,2;4,2) \]
\[ Q_{23} = c(1,1;4,3) \cdot c(2,1;4,3) \cdot c(3,2;4,3) \]

the **Q_{3k}'s** associated to **P_3**

\[ Q_{31} = c(1,1;4,1) \cdot c(2,1;4,1) \cdot c(3,3;4,1) \]
\[ Q_{32} = c(1,1;4,2) \cdot c(2,1;4,2) \cdot c(3,3;4,2) \]
\[ Q_{33} = c(1,1;4,3) \cdot c(2,1;4,3) \cdot c(3,3;4,3) \]

and so on as concerns the others **Q_{ijk}**'s.

It is easy to verify that any product **P_i \cdot Q_{jk}** is a prime implicant of **CF(4)** and all the prime implicants of **CF(4)** are implemented by products **P_i \cdot Q_{jk}**'s. However, no product **P_i \cdot Q_{jk}** with \( i \neq j \) is an implicant of **CF(4)** and, therefore, the considered \( \sum P_i \cdot \sum Q_{jk} \) is not a correct **PCA**.

For example, the product

\[ P_1 \cdot Q_{21} = c(1,1;2,1) \cdot c(1,1;3,1) \cdot c(2,1;3,1) \cdot c(1,1;4,1) \cdot c(2,1;4,1) \cdot c(3,2;4,1) \]

is not an implicant of **CF(4)**.
Four solutions can be adopted to solve this problem.

1. To multiply \( Q_{21} \) by a compatibility or a product of compatibilities.  
   For example:
   \[
   Q'_{21} = Q_{21} \ast c(3,1;4,1)
   \]
   This choice implies the reduction of the merit of \( P_2 \ast Q_{21} \) to the extent of one half.

2. To multiply \( P_1 \) by a compatibility or a product of compatibilities.  
   For example:
   \[
   P'_1 = P_1 \ast c(3,1;4,1)
   \]
   This choice implies the reduction of the merits of \( P_1 \ast Q_{12} \) and \( P_1 \ast Q_{13} \) to the extent of one half.

3. To multiply \( Q_{11} \) by a complemented compatibility or a product of complemented compatibilities.  
   For example:
   \[
   Q'_{21} = Q_{21} \ast !c(1,1;3,1)
   \]
   This choice implies the reduction of the merit of \( P_2 \ast Q_{21} \) to the extent of one half.

4. To multiply \( P_1 \) by a complemented compatibility or a product of complemented compatibilities.  
   For example:
   \[
   P'_1 = P_1 \ast !c(3,2;4,1)
   \]
   This choice implies the reduction of the merits of \( P_1 \ast Q_{11} \), \( P_1 \ast Q_{12} \) and \( P_1 \ast Q_{13} \) to the extent of one half.

Similar considerations can be applied to all the products \( P_i \ast Q_{jk} \) with \( i \not\equiv j \). These considerations prove a quickly decreasing value of the whole merit of the considered PCA.

In order to discuss this problem in more general terms, consider the case of \( P_i \), \( P_j \) and \( Q_{ik} \) under the hypothesis that the numbers of variables involved in \( P_i \) and \( P_j \) are \( m_i \) and \( m_j \) respectively (with \( m_i \leq m_j \)) and that \( n \cdot m_j \) is an increasing function of \( n \).

For the sake of simplicity, without any loss of generality, consider the following example.
\[
\begin{align*}
P_i &= c(1,1;2,1) \ast c(1,1;3,1) \\
P_j &= c(1,1;2,1) \ast c(1,1;3,2) \\
Q_{ik} &= c(1,1;4,1) \ast c(1,1;5,1) \ast c(1,1;6,1) \ast c(2,1;4,1) \ast c(2,1;5,1) \ast c(2,1;6,1) \ast c(3,1;4,1) \ast c(3,1;5,1) \ast c(3,1;6,1) \ast c(4,1;5,1) \ast c(4,1;6,1) \ast c(5,1;6,1)
\end{align*}
\]

In order to transform \( P_j \ast Q_{ik} \) into an implicant of core function \( CF(6) \) one can adopt the above described rule 1 consisting in adding a suitable product of compatibilities to \( Q_{ik} \) as follows.
\[
Q'_{ik} = Q_{ik} \ast c(3,2;4,1) \ast c(3,2;5,1) \ast c(3,2;6,1)
\]
Such an operation implies a reduction of the merit of $P_i * Q_{ik}$ to the extent of $1/2^{(n-m)} = 1/8$ where $m = m_i = m_j = 3$.

In this example only one of variables of $P_1$ (variable 3,1) is different from a variable of $P_2$ (variable 3,2). It is easy to verify that, if the number of variables different in $P_1$ and $P_2$ increases, the merits of $P_i * Q_{ik}$ and $P_j * Q_{jk}$ more quickly decrease.

It follows that by applying previous rule 1 origin $Q'_{ik}$ gives no valuable contribution and can be ignored.

Now consider again the above stated problem of $P_1$, $P_2$ and $Q_{ik}$ and assume that it is solved by applying rule 2.

In this case

$$P'_1 = P_1 * c(3,2;4,1) * c(3,2;5,1) * c(3,2;6,1)$$

This choice implies the reduction of the merits of all the $Q_{jk}$'s, with the exception of $Q_{j1}$, by different orders of magnitude. It is easy to prove the following general relation:

$$\text{merit}(P'_1 * \sum_k Q_{jk}) = \text{merit}(P_1 * \sum_k Q_{jk}) * (2/3)^p$$

where $p=n-m_1$. (The proof is easy but long and it is omitted here for the sake of brevity).

It follows that the merit of $P_1 * \sum_k Q_{jk}$ can increase as $3^p \cdot \text{NMT1}(n)$, but the merits of all the other $P_j * \sum_k Q_{jk}$ (with $j<>i$) can increase only as $2^p \cdot \text{NMT1}(n)$. If the merit of $\text{PCA}_{\text{MAX}}$ must increase as $3^n \cdot \text{NMT}(n)$, $P_1 * \sum_k Q_{jk}$ alone must increase as $3^n \cdot \text{NMT1}(n)$. Indeed, the contributions of the other subsystems $P_j * \sum_k Q_{jk}$ (with $j<>i$) are not sufficient.

The above stated properties make reference to subsystems $<P_i * \sum_k Q_{jk}>$ characterized by the fact that the number $m_i$ of variables involved in $P_i$ does not increase as $n$. Indeed, if $n-m_i$ is a constant, the merit of the considered subsystem increases as $K \cdot \text{NMT1}(n)$, where $K$ is a constant and it appears to be very small with respect to the objective $3^n \cdot \text{NMT1}(n)$. Therefore, it can be ignored unless the number of such subsystems is of the order of $3^n$.

However, also if the number of such subsystems increases as $3^n$, it is easy to prove, by applying the above stated properties, that the total merit of their sum is negligible.

Rule 3 and rule 4 are more effective from the point of view of the levels of merit which can be reached. However, the following property can be proved.

The number of subsystems $<P_i * \sum_k Q_{jk}>$ of $\text{PCA}_{\text{MAX}}$ cannot increase according an exponential law.

A formal proof of the above analyzed property can be developed as follows.

Let $N_S$ be the number of subsystems generated by selecting a number of $P_i$'s and associated $Q_{ij}$'s.

Let $N_C$ be the number of correction PoC's, that is, sequences of variables of $\text{CF}(n)$ containing at least one complemented variable necessary in order that $P_i * Q_{jk}$ is equal to 0 if $i <> j$. It is apparent that any subsystem, with a single exception, must be characterized by at least one correction PoC and that a subsystem must be characterized by at least one correction PoC different from the correction PoC's of the other subsystems. Therefore,

$$N_C >= N_S \cdot (N_S - 1) / 2 - 1$$

It is also apparent that the length $L$ of the longest correction PoC must be such that

$$2^L - 1 >= N_C$$
Now assume that the number of subsystems increases according an exponential law of the type:

\[ N_S = Y \cdot h^n \]

From this assumption the following equations derive:

\[ N_C = Y \cdot h^n \cdot (Y \cdot h^n - 1) / 2 \approx Y^2 \cdot h^{2n} \]

\[ L \approx \log_2 N_C \approx \log_2 (Y^2 \cdot h^{2n}) \approx 2 \log_2 Y + 2n \log_2 h \]

Because of the presence of L complemented variables in the correction PoC, the number of minterms covered by a subsystem is reduced of a factor equal to about \(1 / (2^L)\). Since L is less than the half of the number \(3 \cdot n \cdot (n - 1) / 2\) of variables, it is easy to prove that the number of minterms covered by the sum of all the subsystems is less than

\[ 1 / (L/2) \cdot \text{NMT(ECF(n))} \approx 1 / (Y \cdot h^n) \cdot \text{NMT(ECF(n))} \]

Therefore, the number of minterms covered by the sum of all the subsystems increases as

\[ (3n/h^n) \cdot \text{NMT(ECF(I1,n))} \]

against the hypothesis that \(\text{NMT(PCA_{MAX})}\) increases as

\[ 3^n \cdot \text{NMT(ECF(I1,n))} \]
# Table of Contents

1. Introduction ................................................................................................................................. 1
2. The Core Function .......................................................................................................................... 3
3. A Theorem of Boolean Monotonic Functions ................................................................................. 6
   3.1. Theorem ................................................................................................................................. 6
4. Properties of Core Function ........................................................................................................... 8
   4.1. Property 1 ............................................................................................................................... 8
   4.2. Property 2 ............................................................................................................................... 8
   4.3. Property 3 ............................................................................................................................... 8
5. Products of Compatibilities ........................................................................................................... 9
   5.1. Definition of spurious compatibilities pair .............................................................................. 9
   5.2. Definition of spurious products of compatibilities ................................................................. 9
   5.3. Definition of impure products of compatibilities ..................................................................... 9
   5.4. Definition of core of a PoC .................................................................................................... 9
   5.5. Definition of mark ................................................................................................................. 9
   5.6. Definition of spurius mark ..................................................................................................... 10
   5.7. Definition of extended prime implicants .............................................................................. 10
   5.8. Definition of remainder ......................................................................................................... 10
   5.9. Property 4 ............................................................................................................................. 11
   5.10. Property 5 ............................................................................................................................ 11
   5.11. Property 6 .......................................................................................................................... 11
6. The External Core Function .......................................................................................................... 12
   6.1. Theorem 1 .............................................................................................................................. 12
   6.2. Theorem 2 .............................................................................................................................. 12
   6.3. Theorem 3 .............................................................................................................................. 13
   6.4. Theorem 4 .............................................................................................................................. 13
   6.5. Theorem 5 .............................................................................................................................. 14
   6.6. Theorem 6 .............................................................................................................................. 14
7. The Reference Architecture ........................................................................................................... 15
8. Synthesis of Maximum Merit PCA ............................................................................................... 17
9. The Synthesis of the Most Effective Subsystem ........................................................................... 19
10. The Synthesis of a Complete Subsystem .................................................................................... 21
11. Conclusion .................................................................................................................................. 23
12. Appendix 1 .................................................................................................................................. 24
   12.1. Proof of Theorem 5 ............................................................................................................... 24
   12.2. Proof of Theorem 6 ............................................................................................................... 24
   12.3. Evaluation of K(n) ............................................................................................................... 25
13. APPENDIX 2.........................................................................................................................................27
14. Table of contents........................................................................................................................................31
15. REFERENCES .........................................................................................................................................33
15. REFERENCES


29. A.R.Meo, On the minimal implementation of a monotonic Boolean function, to appear.
