On a Proof of Inequality of the Classes of Decision Problems P and NP

Angelo Raffaele Meo*

Department of Control and Computer Engineering, Politecnico di Torino, Turin, Italy

Review Article

Received: 13-Apr-2024, Manuscript No. GRCS-24-132206; **Editor assigned:** 15-Apr--2024, Pre QC No. GRCS-24-132206 (PQ); **Reviewed:** 29-Apr-2024, QC No. GRCS-24-132206; **Revised:** 06-May-2024, Manuscript No. GRCS-24-132206(R); **Published:** 13-May-2024, DOI: 10.4172/2229-371X.15.1.003

***For Correspondence:**
Angelo Raffaele Meo, Department of Control and Computer Engineering, Politecnico di Torino, Turin, Italy
E-mail: angelo.meo@polito.it

Citation: Meo AR. On a Proof of Inequality of the Classes of Decision Problems P and NP. J Glob Res Comput Sci. 2024;15:003

ABSTRACT

This paper is a new version of three papers presented to the academy of sciences of Turin in 2016 and to the Journal of Computer Science in 2020 and 2022 [1-3]. According to the Journal of Computer Science, more than 6000 readers have "viewed" the two papers published by that journal and more than 1700 readers have downloaded them.

The theorems presented in those papers were based on the equivalence of a deterministic turing machine M processing some input x belonging to $\{0,\mathbf{1}\}^n$ with an n-input Boolean circuit C_n, and on the hypothesis that the number of gates, or AND, OR, NOT operators, appearing in circuit C_n, is polynomial in the running time of the corresponding turing machine.

According to some readers that hypothesis of the equivalence "turing machine -boolean circuit" is questionable. The purpose of this paper is to present a new proof of the inequality of P and NP which is not based on this equivalence. Besides, this new paper contains the answers to the questions asked by some readers and the proofs of some theorems which had been omitted for the sake of brevity.

The analysis discussed in this paper and in its previous versions is based on a well-known NP-complete problem which is called "satisfiability problem" or "SAT". From SAT a new NP-complete problem, called "Core function", derives; this problem is described by a boolean function of the number of the clauses of SAT. In this paper a proof is presented according which the number of boolean operations of the minimal implementation of core function increases with n exponentially. Since the synthesis of core function is an NP-complete problem, this result can be considered as the proof of the theorem which states that the class P of all the decision problems which can be solved in polynomial time does not coincide with the class NP of the problems for which an answer can be verified in polynomial time.

Keywords: Computer science; Polynomial; Decision; Synthesis; Core function; Theorems; Circuit; Operators

INTRODUCTION

Definitions

A brief description of the definitions and properties well known among the scientists of modern computational complexity theory 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 check (f) analysing 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-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 output values.

The importance of NP-completeness derives from the fact that, if we find a polynomial time algorithm for just one NP-complete problem, then we can construct polynomial time algorithms for all the problems in NP and, conversely, if any single NP-complete problem does not have a polynomial time algorithm, then no NP-complete problem has a polynomial time solution. The analysis discussed in this paper will be based on the following well-known NP- complete problem which is called "satisfiability problem or SAT". Given a Boolean expression containing only the names of variables (some of which may be complemented), the operators AND, OR and NOT, and parentheses, is there an assignment of TRUE or FALSE values to the variables which makes the entire expression 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 analysed expressions will be of the type:

$$\begin{aligned}
&3\text{SAT}(t) = \\
&(x_{11} \text{OR } x_{12} \text{ OR } x_{13}) \text{ AND} \\
&(x_{21} \text{OR } x_{22} \text{ OR } x_{23}) \text{ AND} \\
&\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots \\
&(x_{t1} \text{OR } x_{t2} \text{ OR } x_{t3}) \quad \ldots\ldots\ldots\ldots\ldots\ldots\ldots(1)
\end{aligned}$$

where:

***t* is the number of clauses or triplets;**

each x_{ij} is a variable in complemented or uncomplemented form;

each variable may appear multiple times in that expression.

The synthesis of the state of art of question **PvsNP** can be found [4-5].

LITERATURE REVIEW

The core function

The computation of satisfiability problem described by Equation 1 can be decomposed into two processing layers called "compatibility layer" and "core layer".

Compatibility 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 variable,

or

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

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

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

The input variables of one of these operations will be the sign S_{ij} and the binary code $< n_{ij1} n_{ij2} \dots n_{ijm} >$ of the name of variable j of triplet i, and the sign shk and the binary code $< n_{hk1} n_{hk2} \dots n_{hkm} >$ of the name of variable k of triplet h. The output variable of this operation $c(i, j; h, k)$ will be TRUE when, and only when, the two input variables are compatible between themselves.

Therefore, the function implemented by one of the operations of compatibility layer may be written as follows (by using the symbols $*$, $+$, and ! for representing AND, OR and NOT operators, respectively):

$$\begin{aligned} c(i, j; h, k) = {} & S_{ij} * S_{hk} + ! S_{ij} * ! S_{hk} + \\ & + n_{ij1} * ! n_{hk1} + ! n_{ij1} * n_{hk1} + \\ & + n_{ij2} * ! n_{hk2} + ! n_{ij2} * n_{hk2} + \\ & \dots\dots\dots\dots\dots\dots \\ & + n_{ijm} * ! n_{hkm} + ! n_{ijm} * n_{hkm} \dots\dots\dots\dots(2) \end{aligned}$$

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

Core layer: 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. 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 CF(t) (or CF(n)). (Indeed, the number t of triplets appearing in Equation 1 plays the role of symbol n used in the standard complexity theory. In the following analysis, we shall use the symbol t when it is necessary to remember the number of triplets and n in the other cases).

The core function can be determined by proceeding as follows.

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

$$< 1i_1 >, < 2i_2 >, \dots, < ti_t > \dots\dots\dots\dots(3)$$

With $i_1, i_2, \dots, i_t \in \{1,2,3\}$

be the indexes <number of triplet, number of the variable in the triplet> of the selected variables. They will be called "characteristic indexes". Let Π^k be the product of all the compatibility variables relative to the k-th selection (3):

$$\begin{aligned} \Pi^k = {} & c(1, i_1; 2, i_2) * c(1, i_1; 3, i_3) \dots \\ & \dots * c(t - 1, i_{t-1}; t, i_t) \dots\dots\dots\dots(4) \end{aligned}$$

The core function can be defined as the sum

$$\Sigma_k \Pi^k \dots\dots\dots\dots(5)$$

of the products (4) relative to all the selections (3).

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

$$\begin{aligned} CF(3) = {} & c(1,1; 2,1) * c(1,1; 3,1) * c(2,1; 3,1) + \\ & c(1,1; 2,1) * c(1,1; 3,2) * c(2,1; 3,2) + \\ & c(1,1; 2,1) * c(1,1; 3,3) * c(2,1; 3,3) + \end{aligned}$$

$$c(1,1;2,2) * c(1,1;3,1) * c(2,2;3,1) +$$
$$\ldots(\text{other 22 products})\ldots +$$
$$c(1,3;2,3) * c(1,3;3,3) * c(2,3;3,3) \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots (6)$$

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

Notice that the processing work of the elementary operations of compatibility layer (Equation 2) increases as a logarithmic function P(t) of the number of the variables since the increment of the length of the code of a 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 compatibility operations.

Besides, the problem solved by the core layer is clearly in NP, because it is easy to verify a witness solution. It follows that, since the compatibility layer polynomially reduces an NP-complete problem (3SAT) to the problem solved by the core layer, the core function describes a new NP-complete problem. Notice that the boolean function implemented by the core layer may be an incompletely specified function. Indeed, assume that $c(i,j;l,m) = 0$ and $c(i,j;p,q) = 0$. This implies that variable $< i,j >$ and variable $< l,m >$ have the same name and a different sign; similarly, $< i,j >$ and $< p,q >$ have the same name and a different sign. It follows that $< l,m >$ and $< p,q >$ have the same name and the same sign. Therefore, $c(l,m;p,q)$ cannot be equal to 0. Therefore, all the minterms implying $!\,c(i,j;l,m) * !\,c(i,j;p,q) * !\,c(l,m;p,q)$ are incomplete specifications of the Boolean function implemented by the core layer. Some properties of core function have been discussed [7-9].

A theorem of boolean monotonic functions

Let $f(x_1, x_2 \ldots, x_h)$ be an isotonic Boolean function, that is a Boolean function which can be implemented with only AND and OR gates, applied to uncomplemented literals $x_1, x_2 \ldots, x_h$. It was believed that the minimum cost implementation of $f(x_1, x_2 \ldots, x_h)$ always contains only OR and AND gates, but A.Razborov proved that there are isotonic functions whose minimum cost implementation contains also NOT gates [6].

However, there is on 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.

Theorem 4.1: Let I_{min} be one of the minimum cost implementations of the isotonic Boolean function $f(x_1, x_2 \ldots, x_h)$, the cost being defined as the total number of AND, OR or NOT gates. Let C_{min} be the cost of I_{min}.

There exists always an implementation J of f containing only AND and OR gates (in addition, if necessary, to the NOT operators producing input variables $!x_1, !x_2 \ldots, !x_h$) such that $cost(J) <= 2 \cdot C_{min} + h$. Where h is the number of variables.The proof of this theorem can be found [2]. This theorem will be used to simplify the analysis of core function circuits.

Properties of core function

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

Property 1: As stated in section 2, core function is a not completely specified function.

Property 2: Any product defined by Equation 4 is a prime implicant of core function (that is, a product of compatibilities ("PoC") which implies core function and no other term of it).

Property 3: Since the different selections of each of variables defined by Equation 3 are 3, the number of prime implicants of 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 $t \cdot (t-1)/2$ compatibilities.

Products of compatibilities

In the next chapters, reference will be made to the following definitions related to PoC's or "products of compatibilities".

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 $(h = p \text{ and } k \neq q)$ or $(h = r \text{ and } k \neq s)$ or $(l = p \text{ and } m \neq q)$ or $(l = r \text{ and } m \neq s)$.

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

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) which contains one or more complemented variables, will be defined as an impure term of CF. A product of compatibilities which is neither spurious nor impure will be defined as a pure product of compatibilities.

Definition of mark: Consider a pure product of compatibilities satisfying the property that all the indexes of triplet $\{1,2,\ldots,t\}$ appear at least once in some variable. The product of the variables of such a subset will defined as a "mark" or "pure mark" of the prime implicant of which it contains a subset of compatibilities.

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

$$M = c(1,a;2,b) * c(1,a;3,c) * c(1,a;4,d) \ldots\ldots(7)$$

(Where the indexes of triplet are elements of the set $\{1,2,3,4\}$ and a, b, c, d are elements of $\{1,2,3\}$) is a mark of the prime implicant

$$P = c(1,a;2,b) * c(1,a;3,c) * c(1,a;4,d) * c(2,b;3,c) * c(2,b;4,d) * c(3,c;4,d) \ldots(8)$$

since all the indexes of triplet appear at least once in Equation 7.

A mark M can be defined as a "strong mark" if there is a PoC R which is not a mark such that $M * R$ is a prime implicant of core function. For example, the strong mark $c(1,1;2,1) * c(1,1;3,1) * c(1,1;4,1)$, multiplied by $c(2,1:3,1) * c(2,1;4,1) * c(3,1;4,1)$, becomes a prime implicant of CF(4).

If a mark can not be defined as a strong mark, it will be called "a weak mark". For example, the weak mark $c(1,1;2,1) * c(1,1;3,1) * c(2,1;4,1)$, must be multiplied by the weak mark $c(1,1:4,1) * c(2,1;3,1) * c(3,1;4,1)$ in order to produce a prime implicant.

Definition of spurious mark: A spurious PoC in which all the indexes of triplet appear uncomplemented 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 example, in the case of CF(3), $c(1,1;2,1) * c(1,1;3,1) * c(1,1;2,2)$ is a spurious mark of both the prime implicants $c(1,1;2,1) * c(1,1;3,1) * c(2,1;3,1)$ and $c(1,1;2,2) * c(1,1;3,1) * c(2,2;3,1)$. An impure PoC containing a (possibly spurious) mark will be a defined as a (possibly spurious) impure mark.

Definition of remainder: A PoC which is not a mark will be called a "remainder". Also a remainder may be pure (if for any triplet index there is only one uncomplemented index of variable in that triplet) or spurious or impure. A remainder is implied by more than one prime implicant. For example,

$$R = c(2,1;3,1) \text{ of CF(3)}$$

is implied by the following prime implicants

$$\begin{aligned} P1 &= c(1,1;2,1) * c(1,1;3,1) * c(2,1;3,1) \\ P2 &= c(1,2;2,1) * c(1,2;3,1) * c(2,1;3,1) \\ P3 &= c(1,3;2,1) * c(1,3;3,1) * c(2,1;3,1) \end{aligned} \quad \text{.....................................(9)}$$

On the definitions of mark and remainder the following property is based.

Property 1: Let P_1 and P_2 be two PoC's such that $P_1 * P_2$is equal to a prime implicant P of core function. It is easy to prove that either P_1 or P_2 is a mark of P.

Property 2: The product of two marks M_1 and M_2 which are implied by two different prime implicants of core function is not implied by core function.

For example, both $M_1 = c(1,1;2,1) * c(1,1;3,1)$ and $M_2 = c(1,1;2,1) * c(1,1;3,2)$ are marks of CF(3) and are implied by CF(3), but their product is not implied by CF(3). Indeed, for example, the PoC $c(1,1;2,1) * c(1,1;3,1) * !c(1,1;3,2)$ is implied by M_1 but it is not implied by M_2 , by $M_1 * M_2$ and by CF(3).

It follows, as is easy to verify, that the product $(a + b) * (c + d)$ (where a,b,c,d are products of compatibilities) can produce only one or two prime implicants.

Indeed, assume that $a * c$ is a prime implicant P_1 and a is a mark. It follows that $a * d$ cannot be a prime implicant P_2 different from P_1. $b * c$ can be a new prime implicant P_2 only if c is a remainder satisfying the conditions of the following property 7.

Property 7: The best way to integrate operations OR and AND is suggested by the ability of a remainder R to produce 3^m prime implicants, where m is the number of triplet indexes missing in R, through the multiplication by a suitable OoM ("Or Of Marks") . For example, the remainder $R = c(2,1;3,1)$ of CF(4) can be used to obtain the following prime implicants of CF(4):

$$\begin{aligned} &R * OoM = \\ &c(1,1;2,1) * c(1,1;3,1) * c(1,1;4,1) * c(2,1;3,1) * c(2,1;4,1) * c(3,1;4,1) + \\ &c(1,2;2,1) * c(1,2;3,1) * c(1,2;4,1) * c(2,1;3,1) * c(2,1;4,1) * c(3,1;4,1) + \\ &c(1,3;2,1) * c(1,3;3,1) * c(1,3;4,1) * c(2,1;3,1) * c(2,1;4,1) * c(3,1;4,1) + \\ &c(1,1;2,1) * c(1,1;3,1) * c(1,1;4,2) * c(2,1;3,1) * c(2,1;4,2) * c(3,1;4,2) + \\ &c(1,2;2,1) * c(1,2;3,1) * c(1,2;4,2) * c(2,1;3,1) * c(2,1;4,2) * c(3,1;4,2) + \\ &c(1,3;2,1) * c(1,3;3,1) * c(1,3;4,2) * c(2,1;3,1) * c(2,1;4,2) * c(3,1;4,2) + \\ &c(1,1;2,1) * c(1,1;3,1) * c(1,1;4,3) * c(2,1;3,1) * c(2,1;4,3) * c(3,1;4,3) + \\ &c(1,2;2,1) * c(1,2;3,1) * c(1,2;4,3) * c(2,1;3,1) * c(2,1;4,3) * c(3,1;4,3) + \\ &c(1,3;2,1) * c(1,3;3,1) * c(1,3;4,3) * c(2,1;3,1) * c(2,1;4,3) * c(3,1;4,3) \end{aligned} \quad \text{..................... (10)}$$

The external core function

Let I_j be a prime implicant of CF(n). The External Core Function related to I_j, 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 Ik 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 other uncomplemented, implying F).

Of course,

$$\mathrm{ECF}(n, I_j) = I_j * \Pi_{k \neq j}(!\, I_k)$$

where all the prime implicants of Core Function are involved and $!\, I_k$ denotes the complement of I_k (i.e., NOT I_k).

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

$$\mathrm{ECF}(n) = \Sigma_j \mathrm{ECF}(n, I_j) \text{..............................(11)}$$

The importance of External Core Function derives from the following theorems.

Theorem 1: Let $T = I_j * X$

where I_j is a prime iimplicant and X is a possibly empty PoC. T can also be written as $T = T(I_j)$.

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

Theorem 2: Let T be a term of CF(n) implying two or more than two prime implicants of CF(n): $T = T(I_j, I_k)$.

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

Theorem 3: Let $T = T(I_j) = I_j * X$ be a term of CF(n) which is spurious for a single not complemented compatibility X.

If NMT(F) denotes the number of minterms of boolean function F, the number of minterms of $I_j * X$ contained in $\mathrm{ECF}(n, I_j)$ is

$$\mathrm{NMT}(I_j * X * \mathrm{ECF}(n, I_j)) <= (1/2) \cdot \mathrm{NMT}(\mathrm{ECF}(n, I_j)) \text{...................(12)}$$

However, for large values of n,

$$\mathrm{NMT}(I_j * X * \mathrm{ECF}(n, I_j)) \approx (1/2) \cdot \mathrm{NMT}(\mathrm{ECF}(n, I_j)) \text{.......................(13)}$$

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

Theorem 4: Let $I_J * X_1 * X_2 * \ldots X_m$ be a spurious term characterized by m spurious not complemented compatibilities.

The number of its minterms contained in $\mathrm{ECF}(n, I_j)$ is

$$\mathrm{NMT}(I_J * X_1 * X_2 * \ldots X_m * \mathrm{ECF}(n, I_j)) <= (1/(2^m)) \cdot \mathrm{NMT}(\mathrm{ECF}(n, I_j) \text{.....(14)}$$

However, for large values of n,

$$\mathrm{NMT}(I_J * X_1 * X_2 * \ldots X_m * \mathrm{ECF}(n, I_j)) \approx (1/(2^m)) \cdot \mathrm{NMT}(\mathrm{ECF}(n, I_j) \text{..(15)}$$

Theorem 5: Let $T = T(I_j)$ be an impure term of CF(n) characterized by a single impure variable $(!\, X)$: $T = I_j * (!\, X)$.

For large values of n, the number of minterms of $\mathrm{ECF}(n, I_j)$ contained in T is

$$\mathrm{NMT}\ (I_j * (!\, X) * \mathrm{ECF}(n, I_j) \approx\ (1/2)\ \cdot\ \mathrm{NMT}(\mathrm{ECF}(n, I_j) \text{......................(16)}$$

Theorem 6: Let $T = T(I_j)$ be an impure term of CF(n) characterized by m impure variables: $T = I_j * (!\, X_1) * (!\, X_2) * \ldots (!\, X_m)$. For large values of n, the number of minterms of $\mathrm{ECF}(n, I_j)$ contained in T is

$$\mathrm{NMT}(T * \mathrm{ECF}(n, I_j)) \approx ((1/2)^m) \cdot \mathrm{NMT}(\mathrm{ECF}(n, I_j)) \text{...........................(17)}$$

Notice that $\mathrm{NMT}\ (\mathrm{ECF}(n, I_j))\ = \mathrm{NMT}\ (\mathrm{ECF}(n, I_K))$ for any j and k. It will be called NMT1(n).

The value of a node

Let U be a node of the network implementing core function and let F(U) be the boolean function of compatibilities $c(i, j, h, k)$ implemented by U. Since the subnetwork having U as its input does not contain any NOT gate, we can write:

$$\mathrm{CF} = \mathrm{F(U)} * (\mathrm{x}_1 + \mathrm{x}_2 + . \dots) + \mathrm{y}_1 + \mathrm{y}_2 \dots \text{......................}(18)$$

where $\mathrm{x}_1, \mathrm{x}_2, \dots \mathrm{y}_1, \mathrm{y}_2, \dots$, are products of variables of core function, that is , products of compatibilities. Notice also that every $\mathrm{F(U)} * \mathrm{x}_\mathrm{i}$ and every yj must be an extended prime implicant of core function, that is, it is equal to $\mathrm{P} * \mathrm{X}$ where P is a prime implicant of core function and X is a PoC. As we shall see in some examples, often a single product of compatibilities is sufficient to implement core function according to the following equation:

$$\mathrm{CF} = \mathrm{F(U)} * \mathrm{x} + \mathrm{y}_1 + \mathrm{y}_2 \dots \text{................................}(19)$$

where x is a single compatibility. $\mathrm{x}_1, \mathrm{x}_2, \dots \mathrm{x}, \mathrm{y}_1, \mathrm{y}_2, \dots$, will be called “completion code”.

More than one solution of Equation 18 or Equation 19 can produce core function. However, we are looking for a solution characterized by the following property: The total number of minterms of the External Core Functions $\mathrm{ECF(n)}$ contained in $\mathrm{F(U)} * (\mathrm{x}_1 + \mathrm{x}_2 + \dots)$ or in $\mathrm{F(U)} * \mathrm{x}$ takes the maximum value. By definition, this maximum value will be considered as the value val(U) of the node U or the value val(F(U)) of the boolean function implemented by U.

If U is a strong mark M, the value val(M) will be assumed to be equal to NMT1(n) (unless corrections must be made because of spurious or impure variables in F(U) or x_i).The value val(R) of a remainder R will be assumed to be equal to 0. The reasons for these choices are very simple as follows.

M is implied by the set of minterms of $\mathrm{ECF(n)}$ deriving from the prime implicant of which M is a mark and by no other mimterm of $\mathrm{ECF(n)}$, while R is implied by the minterrms of many prime implicants and by the corresponding subsets of minterms of $\mathrm{ECF(n, I_j)}$. Besides, R must be multiplied by a mark x_i (or x) in order to produce a prime implicant and that mark is generated outside the circuit produced by node U. The merit of that prime implicant will be attributed to an other node V of the network.

The value of a weak mark can be assumed as equal to $(1/2) \cdot \mathrm{NMT1(n)}$. Indeed, a weak mark A must be multiplied by a weak mark B in order to produce a prime implicant $\mathrm{I} = \mathrm{A} * \mathrm{B}$.

Variables $\mathrm{x}, \mathrm{x}_1, \mathrm{x}_2, \dots$ must be remainders. This choice is justified by the fact that remainders are the basic units of all the circuits implementing Core Function while a mark used as a variable x or x_i in Equation 18 and Equation 19 would be produced outside the subnetwork which produces node U.

Boolean function implemented by a node U may be the sum of products of compatibilities which may be marks or remainders. The values of node U for the most important values of addends can be determined as follows. By virtue of the above discussed properties it is easy to prove the following theorem. Consider the sum of two strong marks marks M_1 and M_2. For the sake of simplicity, assume that M_2 can be obtained from M_1 by replacing all the occurrences of a single variable v_1 of M_1 with variable v_2 of M_2. For example, if $\mathrm{M}_1 = \mathrm{c}(1{,}1; 2{,}1) * \mathrm{c}(1{,}1; 3{,}1) * \mathrm{c}(1{,}1; 4{,}1)$ and $\mathrm{M}_2 = \mathrm{c}(1{,}2; 2{,}1) * \mathrm{c}(1{,}2; 3{,}1) * \mathrm{c}(1{,}2; 4{,}1)$ are the two strong marks of CF(4), by assuming the complation code (Equation 21) $\mathrm{x} = \mathrm{c}(2{,}1; 3{,}1) * \mathrm{c}(2{,}1; 4{,}1) * \mathrm{c}(3{,}1; 4{,}1)$, we obtain

$$\mathrm{val}(\mathrm{M}_1) = \mathrm{val}(\mathrm{M}_2) = \mathrm{NMT1}(4)$$

$$\mathrm{val}(\mathrm{M}_1 + \mathrm{M}_2) = \mathrm{val}(\mathrm{M}_1) + \mathrm{val}(\mathrm{M}_2) = 2 \cdot \mathrm{NMT1}(4).$$

This does not happen if both M_1 and M_2 are strong marks but they are not characterized by a single different variable. For example, if $\mathrm{M}_1 = \mathrm{c}(1{,}1; 2{,}1) * \mathrm{c}(1{,}1; 3{,}1) * \mathrm{c}(1{,}1; 4{,}1)$ and $\mathrm{M}_2 = \mathrm{c}(1{,}2; 2{,}2) * \mathrm{c}(2{,}2; 3{,}1) * \mathrm{c}(2{,}2; 4{,}1)$ by assuming in Equation 19 $\mathrm{x}_1 = !\,\mathrm{c}(1{,}2; 2{,}2) * \mathrm{c}(2{,}1; 3{,}1) * \mathrm{c}(2{,}1; 4{,}1) * \mathrm{c}(3{,}1; 4{,}1)$ and $\mathrm{x}_1 = !\,\mathrm{c}(1{,}1; 2{,}1) * \mathrm{c}(1{,}2; 3{,}1) * \mathrm{c}(1{,}2; 4{,}1) * \mathrm{c}(3{,}1; 4{,}1)$, we obtain

$$\mathrm{val}(\mathrm{M}_1) = \mathrm{val}(\mathrm{M}_2) = \mathrm{NMT1}(4)$$

$$\mathrm{val}(M_1 + M_2) = (1/2) \cdot \mathrm{NMT1}(4) + (1/2) \cdot \mathrm{NMT1}(4) = \mathrm{NMT1}(4)$$

Other important examples of sums of two PoC's related to CF(4) are the following ones. If $A = c(1,1; 2,1) * c(1,1; 3,1) * c(1,1; 4,1)$ (strong mark) and $B = c(1,2; 2,1) * c(1,2; 3,1) * c(2,1; 4,1)$ (weak mark) then, by assuming $x = c(2,1; 3,1) * c(2,1; 4,1) * c(3,1; 4,1) * !\,c(1,2; 2,1)$ in Equation 19, we obtain:

$$\mathrm{val}(A) = \mathrm{NMT1}(4)$$

$$\mathrm{val}(B) = 0$$

$$\mathrm{val}(A + B) = (1/2) \cdot \mathrm{NMT1}(4) < \mathrm{val}(A\ ) + \mathrm{val}(B)$$

If $M_1 = c(1,1; 2,1) * c(1,1; 3,1) * c(1,1; 4,1)$ (strong mark) and $R = c(1,1; 2,1) * c(1,1; 3,2)$ (remainder) then, by assuming $x = c(2,1; 3,1) * c(2,1; 4,1) * c(3,1; 4,1) * !\,c(1,1; 3,2)$ we obtain:

$$\mathrm{val}(M_1) = \mathrm{NMT1}(4)$$

$$\mathrm{val}(R) = 0$$

$$\mathrm{val}(M_1 + R) = (1/2) \cdot \mathrm{NMT1}(4) < \mathrm{val}(M) + \mathrm{val}(R)$$

On the basis of previous properties it is easy to prove the following theorem.

Theorem 7: Let $F(U) = P_1 + P_2 + \ldots$, where every P_i may be a mark or a remainder. It is easy to prove that $\mathrm{val}\big(F(U)\big) <= \mathrm{val}(P_1) + \mathrm{val}(P_2) + \ldots$ Besides, it is necessary to remember that the sum of two remainders may be a mark. For example, for CF(4), if $R_1 = c(1,1; 2,1) * c(1,1; 3,1) * !\,c(\ 1,1; 3,2)$, $R_2 = c(1,1; 4,1) * c(1,1; 3,2)$, $M = c(1,1; 2,1) * c(1,1; 3,1) * c(1,1; 4,1)$ then

$$R_1 + R_2 = R_1 + R_2 + M$$

However, all the minterms of M_1 are contained in R_1 or R_2. This holds also for the minterms contained in ECF. Therefore, the new mark gives no contribution to ECF. The following new definitions will be adopted.

The set of the minterms of ECF produced by $F(U) * (x_1 + x_2 + \ldots)$ or by $F(U) * x$ will be called SECF(U) (set of minterms of external core function produced by U). The number of elements in SECF(U) will be the value val(U) of node U or the value val(F(U)) of the boolean function implemented by U.

The value of an OR operation

An n inputs OR operation can be implemented as a set of $(n - 1)$ two inputs OR operations. Therefore, we can restrict our attention to two inputs OR operations.

Consider the operation op(U = A OR B). Let SECF(op(A OR B)) be the set of all minterms of ECF contained in SECF(U) and not contained in SECF(A) or SECF(B). The value val(op(A OR B)) of this operation is defined as the number of elements in SECF(op(A OR B)), i.e. the number of new minterms of ECF created by the operation op(A OR B). Note that val(op(A OR B)), which is the value of this operation, is less than or equal to val(A OR B), which is the value of the node that implements the Boolean function (A OR B).

Let $A = a_1 + a_2 + \ldots$, $B = b_1 + b_2 + \ldots$, where every a_i and every b_j is a mark or a remainder. Obviously, $C = A + B = a_1 + a_2 + \ldots + b_1 + b_2 + \ldots$ By virtue of the properties above listed and summarized by Theorem 7.1 and also by virtue of the property above stated according which the mark produced by the sum of two remainders does not give any new contribution to the minterms of ECF, it is easy to prove the following theorem.

Theorem 8: The value of an OR operation is always equal to 0.

The value of an AND operation, the most powerful AND operation

As in the case of OR operations, an n inputs AND operation can be implemented as a set of $(n-1)$ two inputs AND operations. Therefore, we can restrict our attention to two inputs AND operations.

The value of an AND operation having A and B as its inputs and U as its output can be defined as: the number of minterms of the external core function contained in the definition of the value of U (according to Equation 18 or Equation 19) and not contained in the definitions of A and B.

Since we are interested in identifying the most powerful AND operation, as a first step we shall assume that both F(A) and F(B) are sums of remainders so that both $\mathrm{val(A)}$ and $\mathrm{val(B)}$ are equal to 0 and the value of the considered operation is equal to the value of output U.

The most powerful AND operation of this type $(\mathrm{val(A)} = \mathrm{val(B)} = 0)$ can be identified by proceeding as follows.

1. Let $\mathrm{A} = (\mathrm{a_1 + a_2 + a_3 + \ldots})$ and $\mathrm{B} = (\mathrm{b_1 + b_2 + b_3 + \ldots})$, where all the $\mathrm{a_i}$ and $\mathrm{b_j}$ are remainders.
   A product $\mathrm{a_i} * \mathrm{b_j}$ might become the product of two or more than two marks of different prime implicants. For example, in CF(4), $\mathrm{a_i} * \mathrm{b_j}$ might be equal to $\mathrm{m_1} * \mathrm{m_2} = (\mathrm{c(1,1;2,1)} * \mathrm{c(1,1;3,1)} * \mathrm{c(2,1;4,1)}) * (\mathrm{c(1,2;2,1)} * \mathrm{c(1,2;3,2)} * \mathrm{c(1,2;4,1)})$. The complation code might be equal to $\mathrm{x} = \mathrm{c(1,1;4,1)} * \mathrm{c(2,1;3,1)} * \mathrm{c(3,1;4,1)} * \mathrm{c(1,2;3,2)} * \mathrm{c(2,1;3,2)} * \mathrm{c(2,1;4,1)} * \mathrm{c(3,2;4,1)}$ in such a way that
   $$\mathrm{m_1} * \mathrm{m_2} * \mathrm{x} = \mathrm{I_1} * \mathrm{I_2}$$
   where $\mathrm{I_1}$ is the prime implicant produced by $\mathrm{m_1}$ and $\mathrm{I_2}$ is the prime implicant produced by $\mathrm{m_2}$. Since val $(\mathrm{I_1} * \mathrm{I_2}) = 0$ because no minterm contained in the product of two prime implicants may belong to External Core Function, the product $\mathrm{a_i} * \mathrm{b_j} * \mathrm{x}$ can produce only one prime implicant of CF(n).
2. Consider the product $\mathrm{a_1} * \mathrm{b_1}$ if it is a mark. If $\mathrm{a_1}$ is a remainder, at least one of the t indexes of triplet does not appear in the list of triplet indexes of $\mathrm{a_1}$ because, otherwise, $\mathrm{a_1}$ would be a mark, Let it be i’. For the same reason, at least another triplet index does not appear in the list of triplet indexes of $\mathrm{b_1}$, Let it be j’.
   By example, for CF(4):
   $$\mathrm{a_1} = \mathrm{c(1,1;2,1)} * \mathrm{c(1,1;3,1)} * \mathrm{c(2,1;3,1)}$$
   $$\mathrm{b_1} = \mathrm{c(1,1;4,1)} * \mathrm{c(2,1;4,1)}$$
   $$\mathrm{m_1} = \mathrm{a_1} * \mathrm{b_1} \ldots\ldots\ldots\ldots\ldots\ldots\ldots(20)$$
   Triplet index 4 is missing in $\mathrm{a_1}$; triplet index 3 is missing in $\mathrm{b_1}$. In order that $\mathrm{a_1} * \mathrm{b_1} * \mathrm{x}$ is a prime implicant of CF(4), x must be equal to $\mathrm{c(3,1;4,1)}$.
3. Equation 20 is the example of two remainders whose product is a mark without spurious or impure variables. Obviously, the value of this mark is $\mathrm{NMT1(4)}$. According to Theorems 3 and 6, a mark containing a spurious or impure compatibility has a value equal to about $(1/2) \cdot \mathrm{NMT1(n)}$ while a mark containing m spurious or ·impure compatibilities has a value equal to about $(1/2^{\mathrm{m}}) \cdot \mathrm{NMT1(n)}$.
4. Assume that $\mathrm{a_2} * \mathrm{b_1}$ is equal to a new mark m_2 such that $m_2 * x$ is a new prime implicant of CF(4). We can start by assuming that $\mathrm{a_2}$ is equal to $\mathrm{c(1,2;2,1)} * \mathrm{c(1,2;3,1)} * \mathrm{c(2,1;3,1)}$. Since the optimal completion code x is equal to $\mathrm{c(3,1;4,1)}$ and $\mathrm{a_2}$ cannot contain all the three compatibilities involving $< 1,2 >$, the value of $\mathrm{b_1}$ must be corrected by adding $\mathrm{c(1,2;4,1)}$ to $\mathrm{b_1}$: $\mathrm{b1’} = \mathrm{b1} * \mathrm{c(1,2;4,1)}$ Therefore, $\mathrm{val(a_1} * \mathrm{b_1’)} = (1/2) \cdot \mathrm{NMT1(4)}$, $\mathrm{val(a_2} * \mathrm{b_1’)} = (1/2) \cdot \mathrm{NMT1(4)}$.

No increment of the total value has been obtained by introducing a new mark. In the preceding example remainder a_2 is different from a_1 for a single variable ($< 1{,}1 >$ in a_1, $< 1{,}2 >$ in a_2). Now assume that a_1 and a_2 are different for the values of two variables ($< 1{,}1 > \mathit{and} < 2{,}1 > \mathit{in}\ a_1, < 1{,}2 > \mathit{and} < 2{,}2 > \mathit{in}\ a_2$): $a_2 = c(1{,}2;2{,}2) * c(1{,}2;3{,}1) * c(2{,}2;3{,}1)$.
Since the optimal complation code is $c(3{,}1;4{,}1)$ and a_2 cannot contain all the compatibilities involving $< 1{,}2 >$ and $< 2{,}2 >$, b_1' must become: $b_1' = b_1 * c(1{,}2;4{,}1) * c(2{,}2;4{,}1)$.
Therefore, $\mathrm{val}(a_1 * b_1') = 1/4 \cdot \mathrm{NMT1}(4)$, $\mathrm{val}(a_2 * b_1') = 1/4 \cdot \mathrm{NMT1}(4)$, The total value has been further decreased.

5. Let us return to the example relative to CF(4) described by Equation 20, and assume: $a_2 = c(1{,}2;2{,}1) * c(1{,}2;3{,}1) * c(2{,}1;3{,}1)$.
6. In order to implement the new mark m_2 without reducing the value of m_1 it is necessary to introduce a new remainder $b_2 = c(1{,}2;4{,}1) * c(2{,}1;4{,}1)$ so that

$$m_2 = a_2 * b_2 \text{.............................(21)}$$

7. The result stated in 4 and 5 can be extended as follows.
The most powerful AND gate can be obtained by producing every mark as the product of a remainder a_i by a corresponding remainder b_j, and it is not useful to produce two marks m_1 and m_2 by means of three remainders as follows: $m_1 = a_i * b_j$, $m_2 = a_k * b_j$.
8. However, the products $a_1 * b_2$ and $a_2 * b_1$ are not marks. Therefore, it is necessary to introduce the following corrections:

$$a_1 = c(1{,}1;2{,}1) * c(1{,}1;3{,}1) * c(2{,}1;3{,}1)$$
$$b_1 = c(1{,}1;4{,}1) * c(2{,}1;4{,}1) * c(1{,}1;2{,}1)$$
$$a_2 = c(1{,}2;2{,}1) * c(1{,}2;3{,}1) * c(2{,}1{,}3{,}1) * !\,c(1{,}1;2{,}1)$$
$$b_2 = c(1{,}2;4{,}1) * c(2{,}1;4{,}1) * c(1{,}2;2{,}1) * !\,c(1{,}1;2{,}1)$$
$$\mathrm{val}(m_1 = a_1 * b_1) = \mathrm{NMT1}(4)$$
$$\mathrm{val}(m_2 = a_2 * b_2) = \mathrm{NMT1}(4) \cdot\ (1/2) \text{.................(22)}$$

Notice that $c(1{,}1;2{,}1)$ appears in both a_1 and b_1 and $c(1{,}2;2{,}1)$ appears in both a_2 and b_2. According to the corrections introduced in Equation 22, both the products $a_1 * b_2$ and $a_2 * b_1$ are equal to 0. It is also possible to introduce two spurious compatibilities in order that both $a_1 * b_2 * x$ and $a_2 * b_1 * x$ become prime implicants of core function.
For example,

$$a_1 = c(1{,}1;2{,}1) * c(1{,}1;3{,}1) * c(2{,}1;3{,}1)$$
$$b_1 = c(1{,}1;4{,}1) * c(2{,}1;4{,}1) * c(1{,}2{,}4{,}1)$$
$$a_2 = c(1{,}2;2{,}1) * c(1{,}2;3{,}1) * c(2{,}1{,}3{,}1)$$
$$b_2 = c(1{,}2;4{,}1) * c(2{,}1;4{,}1) * c(1{,}1;4{,}1)$$

In this case,

$$\mathrm{val}(m_1 = a_1 * b_1) = \mathrm{NMT1}(4) \cdot (1/2)$$
$$\mathrm{val}(m_2 = a_2 * b_2) = \mathrm{NMT1}(4) \cdot (1/2) \text{...................(23)}$$

It follows that Equation 22 is better than Equation 23. It is easy to prove that Equation 22 represents the best solution for implementing two marks from the viewpoint of their values.

9. Consider the product $(a_1 + a_2) * (b_1 + b_2)$ relative to CF(4) where $a_1 = c(1,1;2,1) * c(1,1;3,1) * c(1,1;3,2)$, $a_2 = c(1,1;2,2) * c(1,1;3,2) * c(1,1;3,1), b_1 = c(2,1;3,1) * c(2,1;4,1) * c(3,1;4,1) * c(2,2;3,1) * c(2,2;4,1)$ $b_2 = c(2,2;3,2) * c(2,2;4,1) * c(3,2;4,1) * c(2,1;3,2) * c(2,1;4,1)$ with $x = c[1,1] * c[4,1]$. The following four marks of CF(4) are generated:

$$m_1 = a_1 * b_1 \text{ involving variables } ([1,1],[2,1],[3,1],[4,1]),$$
$$m_2 = a_1 * b_2 \text{ involving variables } ([1,1],[2,1],[3,2],[4,1])$$
$$m_3 = a_2 * b_1 \text{ involving variables } ([1,1],[2,2],[3,1],[4,1])$$
$$m_4 = a_2 * b_2 \text{ involving variables } ([1,1],[2,2],[3,2],[4,1])$$

It is easy to verify that $\mathrm{val}(m_1) = \mathrm{val}(m_2) = \mathrm{val}(m_3) = \mathrm{val}(m_4) = (1/8) \cdot \mathrm{NMT1}(4)$, Therefore, the total value of the considered product is $(1/2) \cdot \mathrm{NMT1}(4)$. These values are very small. Therefore, there is no point in continuing this line.

10. By following the same line of reasoning which has made it possible to prove that Equation 22 is the best solution for implementing two marks, it is easy to prove that the best solution for implementing three marks is the following one:

$$a_1 = c(1,1;2,1) * c(1,1;3,1) * c(2,1;3,1)$$
$$b_1 = c(1,1;4,1) * c(2,1;4,1) * c(1,1;2,1)$$
$$a_2 = c(1,2;2,1) * c(1,2;3,1) * c(2,1,3,1) * !c(1,1;2,1)$$
$$b_2 = c(1,2;4,1) * c(2,1;4,1) * c(1,2;2,1) * !c(1,1;2,1)$$
$$a_3 = c(1,3;2,1) * c(1,3;3,1) * c(2,1;3,1) * !c(1,1;2,1) * !c(1,2;2,1)$$
$$b_3 = c(1,3;4,1) * c(2,1;4,1) * !c(1,1;2,1) * !c(1,2;2,1) \text{ ....}(24)$$

The value of this solution is

$$(1 + (1/2) + (1/4)) \cdot \mathrm{NMT1}(4)$$

Equation 24 can be extended with the following algorithm in order to implement the marks of all the nine prime implicants of CF(4) compatible with the conditions that the variables $< 3,2 >, < 3,3 >, < 4,2 >, < 4,3 >$ do not appear in that product and the completion code x takes the value $c(3,1;4,1)$.

$$U = A * B = (a_1 + a_2 + \ldots + a_9) * (b_1 + b_2 + \ldots + b_9) \text{.. (24')}$$

Where

$$a_1 = c(1,1;2,1) * c(1,1;3,1) * c(2,1;3,1)$$
$$b_1 = c(1,1;4,1) * c(2,1;4,1) * c(1,1;2,1)$$
$$a_2 = c(1,2;2,1) * c(1,2;3,1) * c(2,1,3,1) * !c(1,1;2,1)$$
$$b_2 = c(1,2;4,1) * c(2,1;4,1) * c(1,2;2,1) * !c(1,1;2,1)$$
$$a_3 = c(1,3;2,1) * c(1,3;3,1) * c(2,1;3,1) * !c(1,1;2,1) * !c(1,2;2,1)$$
$$b_3 = c(1,3;4,1) * c(2,1;4,1) * !c(1,1;2,1) * !c(1,2;2,1)$$
$$a_4 = c(1,1;2,2) * c(1,1;3,1) * c(2,2;3,1) * !c(2,1;4,1)$$
$$b_4 = c(1,1;4,1) * c(2,2;4,1) * c(1,1;2,2) * !c(2,1;3,1)$$
$$a_5 = c(1,2;2,2) * c(1,2;3,1) * c(2,2;3,1) * !c(1,1;2,2) * !c(2,1;4,1)$$

$$b_5 = c(1,2;4,1) * c(2,2;4,1) * c(1,2;2,2) * !c(1,1;,2,2) * !c(2,1;3,1)$$
$$a_6 = c(1,3;2,2) * c(1,3;3,1) * c(2,2;3,1) * !c(1,1;2,2) * !c(1,2;2,2) * !c(2,1;4,1)$$
$$b_6 = c(1,3;4,1) * c(2,2;4,1) * !c(1,1;2,2) * !c(1,2;2,2) * !c(2,1;3,1)$$
$$a_7 = c(1,1;2,3) * c(1,1;3,1) * c(2,3;3,1) * !c(2,1;4,1) * !c(2,2;4,1)$$
$$b_7 = c(1,1;4,1) * c(2,3;4,1) * c(1,1;2,3) * !c(2,1;3,1) * !c(2,2;3,1)$$
$$a_8 = c(1,2;2,3) * c(1,2;3,1) * c(2,3;3,1) * !c(1,1;2,3) * !c(2,1;4,1) * !c(2,2;4,1)$$
$$b_8 = c(1,2;4,1) * c(2,3;4,1) * c(1,2;2,3) * !c(1,1;2,3) * !c(2,1;3,1) * !c(2,2;3,1)$$
$$a_9 = c(1,3;2,3) * c(1,3;3,1) * c(2,3;3,1) * !c(1,1;2,3) * !c(1,2;2,3) * !c(2,1;4,1) * !c(2,2;4,1)$$
$$b_9 = c(1,3;4,1) * c(2,3;4,1) * !c(1,1;2,3) * !c(1,2;2,3) * !c(2,1;3,1) * !c(2,2;3,1) \text{..................}(25)$$

The value of the AND operation implementing those imarks is

$$(1 + 1/2 + 1/4) \cdot (1 + 1/4 + 1/16) \cdot \mathrm{NMT1}(4)$$

which is slightly less than

$$(1 + (1/2) + (1/4))^2 \cdot \mathrm{NMT1}(4) \text{................................}(26)$$

Equation 25 and Equation 26 can be generalized and the value of the best operation implementing the marks of $3^{(n-2)}$ prime implicants of CF(n) becomes:

$$\mathrm{val1(n)} = (1 + 1/2 + 1/4))^{n-3}.(1 + 1/4 + 1/16) \cdot \mathrm{NMT1(n)}\text{.......}(27)$$

which is slightly less than

$$\mathrm{val2(n)} = (1 + 1/2 + 1/4)^{n-2} \cdot \mathrm{NMT1(n)} \text{...........................}(28)$$

In order to prove that the solution proposed by Equation 24 and extended by Equation 24' is the most powerful one, consider three marks which are different for the value of one and only one triplet index. For example, the three marks $m_7 = a_7 * b_7$, $m_8 = a_8 * b_8$, $m_9 = a_9 * b_9$, which have been defined in Equation 24', are different only for the values in triplet index 1. A set as $\{m_7, m_8, m_9\}$ will be called a "set of connected marks". In this example, in order that $a_7 * b_8 = 0$ and $a_8 * b_7 = 0$, both a_8 and b_8 must contain compatibility $!c(1,1;2,3)$. Therefore, the value of mark m_8 will be multiplied by $1/2$.

In order that $a_7 * b_9 = a_9 * b_7 = a_8 * b_9 = a_9 * b_8 = 0$, both a_9 and b_9 must contain $!c(1,1;2,3) * !c(1,2;2,3)$. Therefore, the value of mark m_9 will be multiplied by 1/4. No other solution makes it possible to reduce the values of m_8 and m_9 by a smaller value.

Every mark m appearing in Equation 24' belongs to t sets of connected marks. It is easy to verify on the data of Equation 24' that all the triplets $\{m_i, m_j, m_k\}$ of connected marks have received the same type of corrections and only those corrections have been applied.

Therefore, we can state that the solution proposed in this paper leads to the best solution and that the maximum value of an AND operation of the type above specified is slightly less than $\mathrm{val2(n)} = (1 + 1/2 + 1/4)^{n-2} \cdot \mathrm{NMT1(n)}$.

In the described algorithms, two different marks $m_i = a_i * b_i$ and $m_j = a_j * b_j$ are characterized by one or two complemented compatibilities. These complemented compatibilities reduce the total value. If we use strong marks, not all these complemented compatibilities are necessary. For example, in the following product, characterized by the use of strong marks:

$$A * B = (a_1 + a_2 + a_3) * (b_1 + b_2 + b_3)$$

where:

$$a_1 = c(1,1;2,1) * c(1,1;3,1) * c(1,1;4,1)$$
$$b_1 = c(2,1;3,1) * c(2,1;4,1) * c(3,1;4,1)$$
$$a_2 = c(1,2;2,1) * c(1,2;3,1) * c(1,2;4,1)$$
$$b_2 = b_1$$
$$a_3 = c(1,3;2,1) * c(1,3;3,1) * c(1,3;4,1)$$
$$b_3 = b_1$$

the three variables of triplet $(a_1 + a_2 + a_3)$ are associated to a single remainder b_1=b_2=b_3 Besides, it has been proved that a sum of three strong marks multiplied by a single remainder may reach the value $3 \cdot \mathrm{NMT1}$. These results lead us to hope that strong marks may be useful in the synthesis of the most powerful AND operation. A possible line of research is presented in the following example devoted to the synthesis of CF(5) by using also strong marks.

$$A * B = (a_1 + a_2 + \ldots + a_{27}) * (b_1 + b_2 + \ldots + b_{27}) \text{.....................(29)}$$

where:

$a_1 = c(1,1;2,1) * c(1,1;3,1) * c(1,1;4,1) * c(1,1;5,1) * c(2,1;3,1) * c(3,1;5,1)$

$b_1 = c(2,1;3,1) * c(2,1;4,1) * c(3,1;4,1) * c(2,1;5,1) * c(3,1;5,1) * c(4,1;5,1)$

$a_2 = c(1,2;2,1) * c(1,2;3,1) * c(1,2;4,1) * c(1,2;5,1) * c(2,1;3,1) * c(3,1;5,1)$

$b_2 = b_1$

$a_3 = c(1,3;2,1) * c(1,3;3,1) * c(1,3;4,1) * c(1,3;\ 5,1) * c(2,1;3,1) * c(3,1;5,1)$

$b_3 = b_1$

$a_4 = c(1,1;2,2) * c(1,1;3,1) * c(1,1;41) * c(1,1;5,1) * c(2,2;3,1) * !\,c(2,1;3,1) * c(3,1;5,1)$

$b_4 = c(2,2;3,1) * c(2,2;4,1) * c(3,1;4,1) * c(2,2;5,1) * c(3,1;5,1) * c(4,1;5,1) * !\,c(2,1;3,1)$

$a_5 = c(1,2;2,2) * c(1,2;3,1) * c(1,2;4,1) * c(1,2;5,1) * !\,c(2,1;3,1) * c(2,2;3,1) * c(3,1;5,1)$

$b_5 = b_4$

$a_6 = c(1,3;2,2) * c(1,3;3,1) * c(1,3;4,1) * c(1,3,5,1) * !\,c(2,1;3,1) * c(2,2;3,1) * c(3,1;5,1)$

$b_6 = b_4$

$a_7 = c(1,1;2,3) * c(1,1;3,1) * c(1,1,4,1) * c(1,1;5,1) * c(3,1;5,1) * !\,c(2,1;3,1) * !\,c(2,2;3,1)$

$b_7 = c(2,3;3,1) * c(2,3;4,1) * c(3,1;4,1) * c(2,3;5,1) * c(3,1;5,1) * c(4,1,5,1) * !\,c(2,1;3,1) * !\,c(2,2;3,1)$

$a_8 = c(1,2;2,3) * c(1,2;3,1) * c(1,2;4,1) * c(1,2;5,1) * c(3,1;5,1) * !\,c(2,1;3,1) * !\,c(2,2;3,1)$

$b_8 = b_7$

$a_9 = c(1,3;2,3) * c(1,3;3,1) * c(1,3;4,1) * c(1,3;5,1) * c(3,1;5,1) * !\ c(2,1;3,1) * !\,c(2,2;3,1)$

$b_9 = b_7$

....

$a_{27} = c(1,3;2,3) * c(1,3;3,1) * c(1,3;4,1) * c(1,3;5,3) * c(3,1;5,3) * !\,oc(2,1;3,1) * !\,c(2,2;3,1) *$
$!\,c(3,1;5,1) * !\,c(3,1;5,2)$

$b_{27} = b_{25}$

Notice that

$A * B = (a_1 + a_2 + \ldots + a_{27}) * (b_1 + b_2 + \ldots + b_{27})$

produces all the prime implicants of CF(5) with the exception of those which include variables $< 3,2 >, < 3,3 >, < 4,2 >$ *or* $< 4,3 >$, exactly as the prime implicants produced by Equation 24'.

Starting from Equation 29 it is easy to prove that

$$\mathrm{val}(A * B) = 3 \cdot (1 + 1/2 + 1/4)^3 \cdot \mathrm{NMT1}(5) \text{..................}(29.2)$$

which can be extended to CF(n) as follows:

$$\mathrm{val}(A * B) = 3 \cdot (1 + 1/2 + 1/4)^{n-3} \cdot \mathrm{NMT1}(n) \text{................}(29.3)$$

It has been shown that the value of the product $(A * B)$ specified by Equation 24 is equal to

$$\mathrm{val}(A * B) = (1 + 1/2 + 1/4)^{n-3} \cdot (1 + 1/4 + 1/16) \cdot \mathrm{NMT1}(n)$$

Since $3 > (1 + 1/4 + 1/16)$, $\mathrm{val}(A * B)$ related to Equation 29 (solution with strong marks) is larger than $\mathrm{val}(A * B)$ related to Equation 24 (solution with only remainders). However, as it is easy to prove, in the solution with strong marks $\mathrm{val}(A)$ is equal to $\mathrm{val}(A * B)$ and, therefore, the value of the AND operation producing $(A * B)$ is equal to 0.

To complete the most powerful AND operation

So far all the new marks contained only $< 3,1 >$ and $< 4,1 >$ of triplet 3 or 4 in the compatibilities used in the synthesis of core function. This condition can be removed in order to increase the value of the considered AND operation.

For example, we might start from Equation 24 and we might add nine new remainders $a_{10} \dots a_{18}$ to $a_1 \dots a_9$ and $b_{10} \dots b_{18}$ to $b_1 \dots b_9$, where the new remainders are obtained by replacing all the appearances of $< 4,1 >$ with $< 4,2 >$. Thus nine new marks and nine new prime implicants will be generated. This solution can be extended in such a way that all the 81 prime implicants of CF(4) are generated. However, the values of many of these new 81 prime implicants are very small because many complemented compatibilities must be introduced in order that all the products $a_i * b_j$ with $i <> j$ become equal to 0. Besides, the theoretical multiplication of every mark by the completion code

$$x = c(3,1;4,1) * c(3,1;4,2) * c(3,1;4,3) * c(3,2;4,1) * c(3,2;4,2) * c(3,2;4,3) * c(3,3;4,1) * c(3,3;4,2) * c(3,3;4,3)$$

strongly would reduce the value of the corresponding prime implicants. A better solution can be obtained by integrating the completion codes in the tables of remainders and by multiplying the new terms produced by complemented compatibilities in such a way that all the products $a_i * b_j$ with $i <> j$ become equal to 0. It is necessary to start from the values of a_i and b_i listed in Equation 24. As a first step, assume that:

for all $i <= 9$

$$a_i' = a_i * c(3,1;4,1)$$
$$b_i' = b_i * c(3,1;4,1);$$

for all $i > 9$ and $<= 18$

replace all the appearances of $< 4,1 >$ with $< 4,2 >$ and apply the following corrections:

$$a_i' = a_{i-9} * c(3,1;4,2) * !c(3,1;4,1)$$
$$b_i' = b_{i-9} * c(3,1;4,2) * !c(3,1;4,1).$$

for all $i > 18$ and $<= 27$

replace all the appearances of $< 4,1 >$ with $< 4,3 >$ and apply the following corrections:

$$a_i' = a_{i-18} * c(3,1;4,3) * !c(3,1;4,1) * !c(3,1;4,2)$$

$$b_i' = b_{i-18} * c(3,1;4,3) * !c(3,1;4,1) * !c(3,1;4,2)$$

it follows that:

$\mathrm{val}\left((a'_1 + a'_2 + \ldots + a'_9 + a'_{10} + \ldots a'_{19} + \ldots) * (b_1' + b_2' + \ldots + b_9' + b_{10}' + \ldots b_{19}' + \ldots)\right) = (1 + 1/2 + 1/4) \cdot \mathrm{val1}(4) = (1 + 1/2 + 1/4) \cdot (1 + 1/2 + 1/4) \cdot (1 + 1/4 + 1/16) \cdot \mathrm{NMT1}(4)$

which is the total value of the $3 \cdot 9 = 27$ prime implicants of CF(4) characterized by the following pairs of indexes: $< 3,1 >, < 4,1 >; < 3,1 >, < 4,2 >; < 3,1 >, < 4,3 >$.

The same line of corrections can be applied in order to find the values of the other $6 \cdot 9 = 54$ prime implicants of CF(4) which are characterized by the following pairs of indexes $< 3,2 >, < 4,1 >; < 3,2 >, < 4,2 >; < 3,2 >, < 4,3 >; < 3,3 >, < 4,1 >; < 3,3 >, < 4,2 >; < 3,3 >, < 4,3 >$.

For example, for all $i > 27$ and $<= 36$, it is necessary to replace all the appearances of $< 3,1 >; < 4,1 >$ with $< 3,2 >, < 4,1 >$ and to apply the following correction:

$$a_i' = a_{i-27} * c(3,2;4,1) * !c(3,1;4,1) * !c(3,1;4,2) * !c(3,1;4,3)$$
$$b_i' = b_{i-27} * c(3,2;4,1) * !c(3,1;4,1) * !c(3,1;4,2) * !c(3,1;4,3)$$

The final result of this line of corrections will be the following equation:

$\mathrm{val}\left((a_1 + \ldots + a_{10} + \ldots + a_{19} + \ldots + a_{28} + \ldots + a_{37} + \ldots + a_{46} + \ldots + a_{55} + \ldots + a_{64} + \ldots + a_{73} + \ldots) * (b_1 + \ldots + b_{10} + \ldots + b_{19} + \ldots + b_{28} + \ldots + b_{37} + \ldots + b_{46} + \ldots + b_{55} + \ldots + b_{64} + \ldots + b_{73} + \ldots)\right) = (1 + 1/2 + 1/4 + 1/8 + 1/16 + 1/32 + 1/64 + 1/128 + 1/256) \cdot (1 + 1/2 + 1/4) \cdot (1 + 1/4 + 1/16) \cdot \mathrm{NMT1}(4) \sim 2 \cdot (1 + 1/2 + 1/4) \cdot (1 + 1/4 + 1/16) \cdot \mathrm{NMT1}(4)$

which can be extended to

$$2 \cdot (1 + 1/2 + 1/4)^{n-3} \cdot (1 + 1/4 + 1/16) \cdot \mathrm{NMT1}(n)$$

The set of solutions for completing the AND operation is discussed in the following section.

The value of the most powerful AND operation

The determination of the exact value of the most powerful AND operation is interesting and important, but it is very difficult. The difficulty derives from the fact that the evaluation of the output of that operation is closely connected with the evaluation of the values of inputs A and B. However, as will be proved, in order to solve the problem PvsNP the exact solution of that question is not necessary.

The solution here proposed derives from Equation 25.

Equation 25 is a solution for CF(4), but it can be easily extended to CF(n), as shown by Equation 27. As already observed, the products $(a_1 + a_2 + \ldots + a_9) * (b_1 + b_2 + \ldots + b_9)$ do not produce all the prime implicants of Core Function. Indeed, the prime implicants containing variables different from those appearing in the completion code x (in our example: $< 3,2 >, < 3,3 >, < 4,2 >, < 4,3 >$) do not appear in the list of prime implicants which have been generated.

A simple solution for producing all the prime implicants of core function is the following one.

First, multiply $a_1, a_2, \ldots a_9$ by $c(3,1;4,1)$.

Then extend the list $(a_1 + a_2 + \ldots + a_9)$ with $(a_{10} + a_{11} + \ldots + a_{18})$ and the list $(b_1 + b_2 + \ldots + b_9)$ with $(b_{10} + b_{11} + \ldots + b_{18})$ in order to obtain all the marks and all the prime implicants containing both variables $< 3,1 >$ and $< 4,2 >$, in addition to the marks and the prime implicants containing variables $< 3,1 >$ and $< 4,1 >$ obtained by the product $(a_1 + a_2 + \ldots) * (b_1 + b_2 + \ldots)$:

$a_{10} = c(1,1;2,1) * c(1,1;3,1) * c(2,1;3,1) * c(3,1;4,2)$

$b_{10} = c(1,1;4,2) * c(2,1;4,2) * c(1,1;2,1)$

$a_{11} = c(1,2;2,1) * c(1,2;3,1) * c(2,1,3,1) * !c(1,1;2,1) * c(3,1;4,2)$

$b_{11} = c(1,2;4,2) * c(2,1;4,2) * c(1,2;2,1) * !c(1,1;2,1)$

……

$a_{18} = c(1,3;2,3) * c(1,3;3,1) * c(2,3;3,1) * !c(1,1;2,3) * !c(1,2;2,3) * !c(2,1;4,2) * !c(2,2;4,2) * c(3,1;4,2)$

$b_{18} = c(1,3;4,2) * c(2,3;4,2) * !c(1,1;2,3) * !c(1,2;2,3) * !c(2,1;3,1) * !c(2,2;3,1)$……………………………….(30)

The product $(a_{10} + a_{11} + \ldots + a_{18}) * (b_{10} + b_{11} + \ldots + b_{18})$ produces all the nine implicants of core function containing only $< 3,1 >$ and $< 4,2 >$ and no other variable of triplets 3 amd 4. Similarly, a new product $(a_{19} + \ldots + a_{27}) * (b_{19} + \ldots + b_{27})$ can produce all the prime implicants of core function containing only $< 3,1 >$ and $< 4,3 >$ of the variables of triplets 3 and 4, while the product $(a_{28} + \ldots + a_{36}) * (b_{28} + \ldots + b_{36})$ can produce all the prime implicants of core function characterized by variables $< 3,2 >$ and $< 4,1 >$, and so on.

In this way all the prime implicants of core function will be produced by the product

$$(a_1 + \ldots + a_{10} + \ldots + a_{19} + \ldots + a_{28} + \ldots + a_{37} + \ldots + a_{46} + \ldots + a_{55} + \ldots + a_{64} + \ldots + a_{73} + \ldots)$$
$$(b_1 + \ldots + b_{10} + \ldots + b_{19} + \ldots + b_{28} + \ldots + b_{37} + \ldots + b_{46} + \ldots + b_{55} + \ldots + b_{64} + \ldots + b_{73} + \ldots) \ldots\ldots(31)$$

where the nine pairs of variables $< 3,1 >, < 4,1 >; < 3,1 >, < 4,2 >; < 3,1 >, < 4,3 >; < 3,2 >, < 4,1 >; < 3,2 >, < 4,2 >; < 3,2 >, < 4,3 >; < 3,3 >, < 4,1 >; < 3,3 >, < 4,2 >; < 3,3 >, < 4,3 >$ are involved.

It is easy to prove that every product as, for example, $(a_{10} + a_{11} + \ldots + a_{18}) * (b_{10} + b_{11} + \ldots + b_{18})$ produces a subset of prime implicants of core function disjoint from the other subsets of prime implicants; that is, a prime implicant produced by a subset does not appear in any other subset. Besides, the value of a subset is the optimal one for that subset of prime implicants. Equation 31 can be used to search for the maximum $\mathrm{val}(A * B)$ but it can not be used to search for the maximum value of the AND operation performing the product $(A * B)$. Indeed, $\mathrm{val}(A) = \mathrm{val}(A * B)$ and, therefore, the value of the AND operation is equal to 0. In order to search for the maximum value of the AND operation, the starting point of the above described algorithm must be corrected. A simple example of the necessary correction is the following one.

multiply a_1, a_3, a_5, a_7, a_9 by $c(3,1;4,1)$

multiply b_2, b_4, b_6, b_8 by $c(3,1;4,1)$

multiply $a_{10}, a_{12}, a_{14}, a_{16}, a_{18}$ by $c(3,1;4,2)$

multiply $b_{11}, b_{13}, b_{15}, b_{17}$ by $c(3,1;4,2)$

multiply $a_{19}, a_{21}, a_{23}, a_{25}, a_{27}$ by $c(3,1;4,3)$

multiply $b_{20}, b_{22}, b_{24}, b_{26}$ by $c(3,1;4,3)$

and so on.

Thus, $\mathrm{val}(A)$ and $\mathrm{val}(B)$ will become relatively small and the value of the AND gate will be slightly smaller than $\mathrm{val}(A * B)$. A very large number of the terms appearing in Equation 31 and its variants must be corrected with complemented compatibilities in order that, for any $i <> j$, $a_i * b_j = 0$. In order to reduce the number of complemented compatibilities we can use some marks of the subset of “strong marks”. As an example, we can start from the algorithm described by Equation 29. That algorithm uses many strong marks as shown by the following lines which are the first lines of the code related to CF(5):

Let

$$A * B = (a_1 + a_2 + \ldots + a_{27}) * (b_1 + b_2 + \ldots + b_{27})$$

where:

$$a_1 = c(1,1;2,1) * c(1,1;3,1) * c(1,1;4,1) * c(1,1;5,1) * c(2,1;3,1) * c(3,1;5,1)$$
$$b_1 = c(2,1;3,1) * c(2,1;4,1) * c(3,1;4,1) * c(2,1;5,1) * c(3,1;5,1) * c(4,1;5,1)$$
$$a_2 = c(1,2;2,1) * c(1,2;3,1) * c(1,2;4,1) * c(1,2;5,1) * c(2,1;3,1) * c(3,1;5,1)$$
$$b_2 = b_1$$
$$a_3 = c(1,3;2,1) * c(1,3;3,1) * c(1,3;4,1) * c(1,3;\ 5,1) * c(2,1;3,1) * c(3,1;5,1)$$
$$b_3 = b_1$$

As shown by those lines, three strong marks (for example a_1, a_2 , a_3) are multiplied by the same remainder $(b_1 = b_2 = b_3)$ in order to produce three different prime implicants of core function. This makes it possible to increase $\mathrm{val}(A * B)$ which was

$$(1 + 1/2 + 1/4)^{n-3} \cdot (1 + 1/4 + 1/16) \cdot \mathrm{NMT1}(n)$$

according to Equation 27 while now it is

$$3 \cdot (1 + 1/2 + 1/4)^{n-3} \cdot \mathrm{NMT1}(n) \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots (32)$$

before the corrections with complemented compatibilities. In section 9 it has been proved that the solution proposed by Equation 24 and extended by Equation 27 is the best solution for the product of two sums of remainders. That proof is based on the analysis of the so-called “connected marks”, which are triplets of marks different from each other only for the values of one index. For example,

$$c(1,1;2,1) * c(1,1;3,1) * c(1,1;4,1)$$
$$c(1,2;2,1) * c(1,2;3,1) * c(1,2;4,1)$$
$$c(1,3;2,1) * c(1,3;3,1) * c(1,3;4,1)$$

are a triplets of connected marks of CF (4).

The proof that the solution described by (25) is the best solution was based on the property that all the connected marks had received the best corrections and that only those corrections had been applied. The proof that the solution above presented, where also strong marks are involved, is the best solution can be developed as follows.

As shown by Equation 11, the sum of three marks $(a_1 + a_2 + a_3)$ multiplied by $(b_1 = b_2 = b_3)$ produces three connected marks without any complemented compatibility.

Analogously,

$$(a_4 + a_5 + a_6) * b_4 \text{ (where } b_4 = b_5 = b_6)$$
$$(a_7 + a_8 + a_9) * b_7 \text{ (where } b_7 = b_8 = b_9)$$
$$\ldots..$$
$$(a_{25} + a_{26} + a_{27}) * b_{25} \text{ (where } b_{25} = b_{26} = b_{27})$$

produce other connected marks without any complemented compatibility.

Instead, the connected marks

$$(a_1 + a_4 + a_7) * (b_1 + b_4 + b_7)$$

have requested the following corrections:

$$a_1 = a_1{}'$$

$$a_4 = a_4{}' * !\ c(2,1;3,1)$$
$$a_7 = a_7{}' * !\ c(2,1;3,1) * !\ c(2,1;4,1)$$

The number of corrections necessary for implementing this sum of remainders is larger than the number of corrections used in the solution involving also strong marks. This is the reason for which $\mathrm{val}(A * B)$ relative to the implementation with strong marks is larger than the implementation with remainders.

Probably, $\mathrm{val}(\mathrm{op}(AANDB))$ of the implementation with remainders is larger than the corresponding value of the implementation with strong marks but we narrow the focus on the implementation with strong marks, because it leads to a more secure solution.

Equation 29.3 specifies the value of variable $C1 = A1\ \mathrm{AND}\ B1$ which produces all the 3^{n-2} prime implicants of core function containing variables $< 3,1 >$ and $< 4,1 >$ and not containing variables $< 3,2 >, < 3,3 >, < 4,2 >, < 4,3 >$.

In order to produce all the prime implicant of core function it is necessary to extend the product A_1 AND B_1 as follows:

$C = (A_1 + A_2 + \ldots + A_9) *)(B_1 + B_2 + \ldots + B_9)$

where

$A_2 * B_2$ produce all the prime implicants containing only variables $< 3,1 >$ and $< 4,2 >$;

$A_3 * B_3$ produce all the prime implicants containing only variables $< 3,1 >$ and $< 4,3 >$;

....

$A_9 * B_9$ produce all the prime implicants containing only variables $< 3,3 >$ and $< 4,3 >$;

Since many complemented compatibilities must be introduced in order that, for $i <> j$, $a_i * b_j = 0$,

$\mathrm{val}(C = A\ \mathrm{AND}\ B) < \mathrm{valmax}(n)$

where

$$\mathrm{valmax}(n) = 9 \cdot 3 \cdot (1 + 1/2 + 1/4)^{n-3} \cdot \mathrm{NMT1}(n)$$

Besides ,

$$\mathrm{val}(\mathrm{op}(A\ \mathrm{AND}\ B) < \mathrm{val}(A\ \mathrm{AND}\ B)$$

CONCLUSION

Since the number of minterms of $\mathrm{ECF}(n)$ contained in CF(n) is equal to $3^n.\mathrm{NMT1}(n)$ and the value of an AND or OR operation, that is the number of new minterms of $\mathrm{ECF}(n)$ produced by an operation, is less than

$$\mathrm{valmax}(n) = 9 \cdot 3 \cdot (1 + 1/2 + 1/4)^{n-3} \cdot \mathrm{NMT1}(n)$$

the number of operations necessary to implement CF(n) is larger than

$$3^n/(9 \cdot 3 \cdot (1 + 1/2 + 1/4)^{n-3})$$

and, therefore, it increases exponentially with n.

Since the synthesis of Core Function CF(n) is an NP-complete problem, this result is equivalent to proving that P and NP do not coincide.

REFERENCES

1. A.R.Meo: “On the P versus NP question”. https://www.accademiadellescienze.it/attivita/editoria/lavori-di-soci

2. A.R.Meo: “On the P vs NP question: a proof of inequality”, *arXiv*:1802.005484.

3. Lance Fortnow: “The status of the P versus NP problem”, Communications of the ACM, September 2009.

4. “The P versus NP problem,” in J.Carlson, A. Wiles (eds.), *The Millennium Prize Problem*, pp.88-104, Providence: American Mathematical Soceity.

5. Razborov, “Lower bounds on the monotone complexity of some Boolean functions”. *Soviet Mathematics-Doklady* 31,(1985)485-493.

6. A.R.Meo: “Some theorems concerning the core function” in “Concurrency, Graphs and Models”, Springer-Verlag Berlin Heidelberg, 2008.

7. S.A.Cook: The complexity of theorem proving procedures, in: Proc. 3rd Annual ACM Symp. on Theory of Computing, (1971),pp.151-158. ACM Press.

8. K.Mulmuley and M.Sohoni. Geometric complexity theory I: An approach to the P vs. NP and related problems. SIAM Journal on Computing, 31(2):496{526,2001}.