DECOMPOSITION OF BOOLEAN FUNCTIONS – RECOGNIZING A GOOD SOLUTION BY TRACES

Arkadij Zakrevskij

Abstract: The problem of sequent two-block decomposition of a Boolean function is regarded in case when a good solution does exist. The problem consists mainly in finding an appropriate weak partition on the set of arguments of the considered Boolean function, which should be decomposable at that partition. A new fast heuristic combinatorial algorithm is offered for solving this task. At first the randomized search for traces of such a partition is fulfilled. The recognized traces are represented by some "triads" - the simplest weak partitions corresponding to non-trivial decompositions. After that the whole sought-for partition is restored from the discovered trace by building a track initialized by the trace and leading to the solution. The results of computer experiments testify the high practical efficiency of the algorithm.

Keywords: Boolean function, non-disjunctive decomposition, appropriate partition, combinatorial search, recognition, randomization, computer experiment.

ACM Classification Keywords: G.2.1 Combinatorics – combinatorial problems, combinatorial search, G.3 Probability and Statistics – randomization.

Introduction

The well-known problem of Boolean functions decomposition consists in replacement, if possible, of the considered function by an equivalent composition of several functions with smaller number of variables. Different sorts of the decomposition were offered, including *sequential two-block decomposition*, called simple decomposition, which could be disjunctive or non-disjunctive.

As a result of the disjunctive decomposition an initial Boolean function $f(x)$ is substituted by a composition $g(h(u), v)$ at a partition u/v on the set of arguments $x = (x_1, x_2, \ldots, x_n)$: $u \cup v = x$, $u \cap v = \varnothing$. [Povarov, 1954], [Ashenhurst, 1959]. In this case the partition u/v is named *strong*.

The more general case is represented by non-disjunctive decomposition, when function $f(x)$ is substituted by a composition $g(h(u, w), w, v)$ at a *weak* partition u/v, when $x = u \cup w \cup v$, $u \cap w = u \cap v = w \cap v = \varnothing$ [Curtis,1962], [Zakrevskij, 1964].

In both cases the conditions $|u| > 1$ and $|v| > 0$ must be satisfied, otherwise the decomposition is trivial – it exists always.

Solving the following two tasks are laying in the basis of decomposition methods.

The task 1. For a given function $f(x)$ and partition u/v (strong either weak) to find out, whether $f(x)$ is decomposable at u/v, i. e. whether there exists a non-trivial composition ($g(h(u), v)$ or $g(h(u, w), w, v)$) equivalent to function f (and, maybe, to find functions g and h).

If such a composition does exist, we shall term partition u/v as *appropriate*.

The task 2. For a given function $f(x)$ to find an appropriate partition.

A lot of papers were devoted to the solution of the first task, far less – to the second one. Indeed, the second task is more difficult. The main attention in this paper is paid just to it.

Solving Task 1

The task of checking a Boolean function $f(x)$ for decomposability at a given partition u/v on the set of arguments x was regarded in [Povarov, 1954], [Ashenhurst, 1959], [Curtis,1962],.where a necessary and sufficient condition for that was found, which could be formulated as in the following assertion:

Assertion 1. Let $f_i(u, v)$ be the coefficients of the Shannon disjunctive decomposition of Boolean function $f(x)$ by variables of set w. Then function $f(x)$ is decomposable at partition u/v, if and only if the coefficients of the similar decomposition of each function $f_i(u, v)$ by variables from set u accept not more than two different values.

Checking Boolean functions for satisfying this condition was laid into the base of many known methods offered for solving task 1.

A new algorithm for solving the same problem is developed, based on representation of considered functions by long, 2^n-component, Boolean vectors and using efficient component-wise operations above them. It is checking a Boolean function for satisfying the certain condition formulated in terms of symmetry operations introduced in [Zakrevskij, 1964] and defined as follows:

$S_{xi}^{\vee} f(x) = f(x_1, \ldots, x_i, \ldots, x_n) \vee f(x_1, \ldots, \neg x_i, \ldots, x_n)$ is disjunctive symmetrization of Boolean function $f(x)$ by argument x_i,

$S_u^{\vee} f(x) = S_{u1}^{\vee}(S_{u2}^{\vee} \ldots (S_{uk}^{\vee} f(x)) \ldots)$ is disjunctive symmetrization of Boolean function $f(x)$ over a given subset $u = (u_1, u_2, \ldots, u_k)$ of set x.

The result of applying operator $S_u^{\vee}$ to function $f(x)$ could be interpreted as follows, in terms of the Boolean space M of variables from set x, where function $f(x)$ is defined, and its subsets. Any interval of M with inner variables taken from set u, which has if only one 1, is filled up with 1s completely.

Operator $S_v^{\vee} f(x)$ is defined in the analogous way – function f(x) is disjunctively symmetrised over set v.

The suggested algorithm checks function $f(x)$ for satisfying the condition formulated in Assertion 1. It takes the initial coefficient f_u^0 of Shannon disjunctive decomposition of function $f(x)$ by set u, then, by using operation $f^+(x) = f(x) \oplus f_u^0$, modifies function $f(x)$ in such a way that all coefficients coinciding with f_u^0 are changed for 0, and finally finds out if all the remaining (now changed and marked with non-zero values of $S_v^{\vee} f^+(x)$) coefficients are equal to each other.

In other words, the algorithm is working in accordance with the following assertion.

Assertion 2. A Boolean function $f(x)$ is decomposable at partition u/v, if and only if

$$f^{++}(x) = (f^+(x) \oplus S_u^{\vee} f^+(x)) \wedge S_v^{\vee} f^+(x) = 0.$$

Two examples are given below demonstrating calculation of function $f^{++}(x)$. In first example the initial function turns out to be decomposable at partition u/v, while in second one the result is negative.

	f	f_u^0	f^+	$S_u^{\vee} f^+$	$f^+ \oplus S_u^{\vee} f^+$	$S_v^{\vee} f^+$	f^{++}
	1001	1001	0000	1000	1000	0000	0000
Example 1	0001	1001	1000	1000	0000	1111	0000
	0001	1001	1000	1000	0000	1111	0000
	1001	1001	0000	1000	1000	0000	0000
	1001	1001	0000	1010	1010	0000	0000
Example 2	0001	1001	1000	1010	0010	1111	0010
	0011	1001	1010	1010	0000	1111	0000
	1001	1001	0000	1010	1010	0000	0000

All initial and intermediate functions are presented in convenient for visual perception form of Boolean 4×4-matrices, which rows correspond to different values of vector $u = (u_1, u_2)$ (does not matter in which order), and columns – to different values of vector $v = (v_1, v_2)$.

So, checking a Boolean function for decomposability at a given partition is reduced to several rather simple operations over 2^n-component Boolean vectors.

Relations on the Set of Partitions. Triads

As was mentioned, the second task is more difficult. A heuristic swift combinatorial algorithm is offered below to solve it. Some preliminary theory should be introduced before its formulation.

Consider two partitions u/v and u^*/v^*, such that $u^* \subseteq u$ and $v^* \subseteq v$. Let's speak, that partition u^*/v^* *submits* to partition u/v. The following assertions are fair by that.

Assertion 3. If the function $f(x)$ is decomposable at partition u/v, it is decomposable as well at partition u^*/v^*. Hence, if the function $f(x)$ is not decomposable at partition u^*/v^*, it is not decomposable also at partition u/v.

Let's assume $|u| = k$ and $|v| = m$. Partition with $k = 2$ and $m = 1$ we shall term as a *triad*. It is the simplest of partitions, at which a non-trivial decomposition can be defined.

Assertion 4. The number of triads is equal to $C_n^2 (n-2) = \frac{n(n-1)(n-2)}{2}$.

Assertion 5. The Boolean function $f(x)$ is not decomposable, if it is not decomposable at any of triads.

It follows from here that the function is decomposable, if and only if it is decomposable if only at one of triads.

Let's estimate the probability of decomposability of a random Boolean function $f(x)$ of n variables. Consider some triad u/v, having put for example $u = (a, b)$ and $v = (c)$, and some arbitrary coefficient $\varphi(a, b, c)$ of function $f(x)$ Shannon decomposition by variables of set $w = x \setminus (u \cup v)$.

Assertion 6. The number of different values of the coefficients of the Shannon decomposition of an arbitrary Boolean function $\varphi(a, b, c)$ by variables a and b does not exceed two, if the coefficients φ_0 and φ_1 of the decomposition of function φ by variable c obey to the condition: $\varphi_0 = \neg\varphi_1$ or at any rate one of the coefficients φ_0 and φ_1 is equal to constant 0 or 1.

It follows from this condition

Assertion 7. Among 256 different Boolean functions $\varphi(a, b, c)$ there are 74 functions, for which the number of different values of coefficients of Shannon decomposition by variables a and b does not exceed two.

Let's designate through γ the share of such functions: $\gamma = 74/256$.

Considering the above assertions and taking into account, that for any triad $|w| = n - 3$ and, therefore, the number of coefficients of the function $f(x)$ Shannon decomposition by w is equal to 2^{n-3}, we shall formulate

Assertion 8. At equiprobable sampling of function $f(x)$ from the set of all Boolean functions of n variables the probability of the function $f(x)$ decomposability is bounded above by the value

$$C_n^2 (n-2) \cdot \gamma^{2^{n-3}}.$$

It is important to note, that this value fast tends to zero with growth of n. For example, at $n = 6, 12, 18, 24$ it receives accordingly values 0.003, 10^{-135}, 10^{-8827}, $10^{-565200}$.

It is obvious from this that decomposability of arbitrary Boolean functions is rather improbable. Therefore, the task of decomposition has practical sense only for such a function, which, as it must be known a priori, is decomposable, and it is necessary only to find the appropriate composition. The task in this setting is considered below.

Search for Traces of an Appropriate Partition

Let's assume, that function $f(x)$ is known, obtained as a result of the composition $g(h(u, w), w, v)$ of two random Boolean functions g and h on a given, also random, weak partition u/v of the set of arguments x. It is required to detect this composition by the analysis of function $f(x)$. This problem can be reduced mainly to finding the appropriate partition u/v.

The elementary way of finding this partition consists in exhaustive search of all nontrivial weak partitions on the set of arguments x (such, that $k > 1$ and $m > 0$) and checking the function $f(x)$ for the decomposability on everyone of these partitions. However such a way is rather time-consuming, what is evident from the following assertion.

Assertion 9. The number of all nontrivial weak partitions on the set of n arguments equals $3^n - 2^{n+1} - n2^{n-1} + n + 1$.

This estimate is obtained by deleting from the set of all three-block partitions (their number equals 3^n) such ones, which do not obey to the condition "$k > 1$ and $m > 0$".

For example, at $n = 6, 12, 18, 24$ this number receives accordingly following values: 416; 498,686; 384,536,924; 282,194,655,482.

Analyzing such big number of partitions one by one is practically impossible. Search for appropriate partition u/v can be accelerated by preliminary random looking for some trace of partition u/v, i. e. some partition u^*/v^* submitting to u/v, after which the latter could be found by corresponding expansion of sets u^* and v^*.

Assertion 10. The number of partitions submitting to partition u/v, is equal to 2^{k+m}. So the random search for partition u/v can be accelerated in 2^{k+m} times.

Let's assume, that during the search q partitions are selected by random and analyzed, until a trace of u/v is found.. Suppose, that each appropriate partition submits to the required partition u/v (another case will be analyzed a little later). At this supposition the following assertion is fair.

Assertion 11. The expectation $M(q)$ of the value q is equal approximately to $3^n / 2^{k+m}$.

For example, at $n = 6, 12, 18, 24$ expectation $M(q)$ receives accordingly values 11.4, 130, 1478, 16834, if $k = m = n/2$ (in that case the formula is simplified to $(3/2)^n$), and values 45; 2,076; 94,577; 4,309,504, if $k = m = n/3$.

Nevertheless, expectation $M(q)$ remains rather big, and that is why a more efficient method is suggested, which is looking for traces between the simplest weak partitions (i.e. triads), as their number is much less. For example, at the same values $n = 6, 12, 18, 24$ the total number of triads according to the assertion 3 receives values 60, 660, 2448, 6072.

Assertion 12. The number of triads submitting to partition u/v, is equal to $C_k^2 m$.

The next assertion follows from here.

Assertion 13. When looking a trace of partition u/v between triads, the expectation $M(q)$ is equal to $C_n^2(n - 2) / C_k^2 m$.

That is too far less, compared with preceding methods. For example, on the same series of values $n = 6, 12, 18, 24$ the expectation $M(q)$ receives accordingly values 30, 27.5, 27.2, 27.1, if $k = m = n/3$ and values 6.7, 7.3, 7.6, 7.7, if $k = m = n/2$. That testifies the essential reducing of the number of checked triads by looking for an appropriate one as well as weak dependence of the run-time on the number of variables, what is very important. Let's remark in this connection, that if $k = m = n/t$, then the ratio representing the value $M(q)$ can be approximated by constant t^3, for example, by constant 27, when $t = 3$, and constant 8, when $t = 2$. These values determine the expected run-time of the search for partition u/v.

False Traces and Side Solutions

Let's remark, that not any appropriate triad submits to the required partition, what is illustrated by results of a conducted experiment, with parameters $n = 6$, $k = 3$, $m = 3$. During this experiment there were generated the random partition $u/v = (a, d, e) / (b, c, f)$ on the set of variables $x = (a, b, c, d, e, f)$ and a couple of random Boolean functions $h(u)$ and $g(v)$. Then the composition $g(h(u), v)$ was constructed and the corresponding Boolean function $f(x)$ obtained, after which all triads on the set x were checked and appropriate ones were selected between them. The discovered appropriate triads are marked by 1s in the following table of triads: for example, in the first row the triads *bd*/*a*, *be*/*a*, *cd*/*a*, *de*/*a* and *df*/*a* are marked un such a way. But it appears that only 9 from 35 appropriate triads submit to partition *u*/*v* - they are marked by the bold font. The remaining 26 triads represent tracks of some side solutions, and that is why they are called *false* triads below.

	a a a a a	*b b b b*	*c c c*	*d d*	*e*
	b c d e f	*c d e f*	*d e f*	*e f*	*f*
a	. . . . .	. 1 1 .	1 . .	1 1	.
b	. . **1 1** .	. . . .	1 . .	**1** 1	.
c	. . **1 1** .	. 1 . .	. . .	**1** 1	.
d	1 1 . 1 1	1 . 1 1	. 1 1	. .	1
e	1 . 1 . .	. 1 . .	1 . .	. 1	.
f	. . **1 1** .	. 1 . .	1 . .	**1** .	.

However, with growth of the number of variables n the quota of false triads promptly diminishes.

Let's estimate the probability of violation of the supposition "any appropriate triad submits to partition *u*/*v*" in the regarded situation of checking function $f(x)$ for satisfying the condition of decomposability at the given partition *u*/*v* and at random sampling of functions g and h in the composition $g(h(u, w), w, v)$.

Each of three units of a current triad can be selected from one of the sets u, v or w. Therefore, there are $3^3 = 27$ situations, which can symbolically be presented as *uu/u, uu/v, uu/w, uv/u, uv/v, uv/w, uw/u, uw/v, uw/w, vu/u, vu/v, vu/w, vv/u, vv/v, vv/w, vw/u, vw/v, vw/w, wu/u, wu/v, wu/w, wv/u, wv/v, wv/w, ww/u, ww/v, ww/w.* Only in one of them (namely in *uu/v*) the triad submits to partition *u*/*v*, and among remaining ones the maximum probability the considered supposition gains in situation *uu/u*.

Assertion 14. The probability that a random triad in situation *uu/u* will appear to be appropriate is equal to $\gamma 2^{n-m-3}$.

For example, if $n = 12$ or $n = 18$, the probability receives accordingly values $2.0 \cdot 10^{-3}$, $1.4 \cdot 10^{-5}$ at $m = n/3$ and values $2.4 \cdot 10^{-2}$, $5.8 \cdot 10^{-4}$ at $m = n/2$.

In remaining situations (excepting uu/v) this probability is no more. It follows from here, that in practical situations (when n exceeds 10) any appropriate triad, most likely, submits to the required partition, therefore, we can relay on Assertion 11 for estimating the time needed to find a true trace.

This conclusion is confirmed by results of two series of computer experiments, during which 10 random compositions for each considered set of parameters n, k, m were generated and any appearance of a false triad was registered. In the first series, where 150 experiments were carried out at 15 sets (6, 2, 2), (7, 3, 2), (8, 3, 3), …, (20, 7, 7), the false triads have appeared only at $n = 6$ and $n = 7$. In the second series (on 15 sets (6, 3, 3), (7, 4, 3), (8, 4, 4), …, (20, 10, 10)) the false triads were met at $n = 6, 7, 8, 9, 10$. At $n = 10$ only one such triad on 10 experiments has appeared.

It follows from here that for finding required partition *u*/*v* at $n \geq 10$ it is enough to expand properly sets u and v, which constitute the detected appropriate triad.

Meanwhile, when n ≤ 10 and a good solution does exist (with $k > 2$ and $m > 1$), false tracks can be eliminated by using the following procedure of initial expansion. One by one we test variables from set x, not coming in the triad, until discover such one, which inclusion into set u^* results in a new (expanded) appropriate value of partition u^*/v^*. If such a variable is not found, we eliminate that triad as false and look for another appropriate triad. When we find such variable, we try similarly to expand set v^*. If the appropriate element misses, we again eliminate the triad and look for another one. If both checks lead to the positive result, the regarded triad is recognized as a true trace and is accepted for the next expansion.

Let's return to reviewing the former example of the set of appropriate triads. Some of its elements are rejected by this procedure because they cannot be expanded by addition of a new element into set u^*. Some other elements safely passed the first trial but not sustained the second one (when trying to expand set v^*), so they are rejected also. In such a way all 26 false tracks are rejected.

Heuristic Search Algorithm

Taking into account the presented reasons, we shall offer the following heuristic algorithm for detection of a good (with $k > 2$ and $m > 1$) partition u/v, at which the considered function $f(x)$ is decomposable.

A. We consider a sequence of triads selected by random. As soon as a current triad happens to be appropriate, we apply the described above procedure of elimination of false traces between appropriate triads. In result we find a perspective appropriate triad and regard it as the initial value of variable partition u^*/v^*.

B. We consider successively all elements of set $x \setminus (u^* \cup v^*)$. Regarding the current element, we include it into set u^*. If the obtained by that extended partition u^*/v^* appears not to be appropriate, the element is deleted from u^* and put into v^*. If the extended partition is not appropriate now, the element is deleted from v^*.

C. The partition u^*/v^* obtained after exhaustive search of all variables is accepted as the solution. Adduced above estimations allow to affirm, that with a high probability it will coincide with the sought-for partition u/v.

In the case of large number of variables (ten or more) the algorithm can be simplified by excluding the procedure of elimination of false traces, because the probability of side solutions becomes negligibly small.

In the case when an appropriate complete partition u/v does exist the search of this partition can be essentially speeded up by looking only for the set v, as set u can be obtained as the complement of set v up to set x.

The set v can be obtained by way of expansion of set v^* in the found appropriate triad by fast iterative algorithm, which includes the current variable x_i into set v, if

$$S_{xi}^{\oplus}(T) = 0,$$

where $S_{xi}^{\oplus}$ is the operator of symmetrization a Boolean function by EXOR-operation, known as differential operator, and

$$T = S_v^{\vee} f^+ = S_v^{\vee}(f(x) \oplus f_u^0).$$

In case of weak partition, when $w \neq \varnothing$, the redundant check of variables from set $x \setminus v$ for possibility of inclusion into set u will be carried out.

Results of Experiments

The offered heuristic algorithm was programmed in C++ (by I. Vasilkova) and tested on computer (Pentium IV, 2.8 GHz). In a series of experiments the parameters n, k, m were fixed, a random partition u/v on the set x and functions g, h were generated, then function $f(x)$ was calculated. After that the above described algorithm was fulfilled, which found partition u/v for the function $f(x)$, the number q of triads scanned by search for traces was ascertained, and the total time t (in seconds) spent during search for the partition was measured. The obtained results are represented in the table below, which right part corresponds to splitting of the set of arguments x in three parts u, w and v, whenever possible the same size, and left part – in two: u and v.

As can be seen from the table, the regarded task of Boolean function decomposition is solved in less than one minute, when the number of arguments does not exceed 26. The amount of memory for representation of Boolean function $f(x)$ grows quickly, reaching 2^{28} = 268 435 456 bits when n = 28. This value is critical for the given experiment because of the restrictions on the used operation memory.

Note that the table begins with n = 14, because $t \leq 0.00$, if $n \leq 14$.

n	*k/m*	*q*	*t*	*k/m*	*q*	*t*
14	7/7	3	0.00	5/5	39	0.00
15	8/7	2	0.00	5/5	78	0.02
16	8/8	1	0.00	6/5	18	0.01
17	9/8	11	0.02	6/6	34	0.05
18	9/9	4	0.02	6/6	42	0.09
19	10/9	3	0.03	7/6	39	0.17
20	10/10	9	0.11	7/7	3	0.16
21	11/10	1	0.11	7/7	3	0.39
22	11/11	3	0.48	8/7	6	2.41
23	12/11	9	2.17	8/8	16	7.05
24	12/12	3	2.48	8/8	26	18.17
25	13/12	4	5.61	9/8	19	33.16
26	13/13	4	11.64	9/9	3	52.11
27	14/13	19	60.13	9/9	36	187.55
28	14/14	19	1280.67	10/9	3	1585.03

Conclusion

A new heuristic algorithm for Boolean function f(x) decomposition g(h(u, w), w, v) is developed, which recognizes the weak partition u/v on the set of arguments x by finding first some trace of it in the form of a triad, and using this trace for restoring the partition u/v as a whole. As computer experiments show, the algorithm is very efficient.

Acknowledgement

The work had been performed thanks to the support of ISTC (Project B-986).

Bibliography

[Povarov, 1954] Povarov G.N. About functional decomposition of Boolean functions. – Reports of the AS of USSR, 1954. – V. 4, No 5 (in Russian).

[Ashenhurst, 1959] Ashenhurst R.L. The decomposition of switching functions. Proc. International Symposium on the Theory of Switching, Part 1. – Harvard University Press, Cambridge, 1959, pp. 75-116.

[Curtis,1962] Curtis H.A. Design of switching circuits. – Van Nostrand, Princeton, N. J., 1962.

[Zakrevskij, 1964] Zakrevskij A.D. Algorithm of a Boolean function decomposition. – Annals of Siberian Physical-Technical Institute, 1964. – V. 44, pp. 5-16 (in Russian).

Author's Information

Arkadij Zakrevskij – The United Institute of Informatics Problems of the NAS of Belarus, Surganov Str. 6, 220012, Minsk, Belarus; e-mail: zakr@newman.bas-net.by