Tools for Calculating Autocorrelation Spectrum by Using The Wiener-Khinchin Theorem

Miloš M. Radmanović

Abstract — **The paper consider tools for calculating autocorrelation spectrum by using the Wiener-Kinchin theorem. The first tool calculates autocorrelation spectrum implemented over matrices, the second tool use fast Walsh transform over matrices and the third tool calculates autocorrelation spectrum implemented over decision diagrams. Then, I discussed "LGSynth93 benchmark" testing statistics for each tool and tool's efficiency. Also, I analyzed time and space complexity for each tool, especially for tool which use calculations over the decision diagrams.**

Keywords — **Switching functions, Autocorrelation spectrum, Wiener-Kinchin theorem, decision diagrams, software tools..**

I. Introduction

Much work has been performed in applying transforms to switching functions in order to achieve a more global view of the function. Transforms such as the Walsh and their applications in digital logic are well researched [6]. Also, there is a lot of software support. There is far less work, however, on the use of other transforms such as the autocorrelation transform [7].

An alternative view of the switching function is the autocorrelation spectrum (all autocorrelation coefficients) of a function. The autocorrelation coefficients of a function are calculated using the autocorrelation function. They provide the measure of the function's similarity to itself, shifted by given amount, Eq. (1).

The autocorrelation coefficients have been used in various areas including optimization and synthesis of combinational logic [1], [4], [5], variable ordering for decision diagrams [2] and compute the estimate $C(f)$ of a function's complexity [3], [1].

However, their use has been limited, likely due to the fact that until recently, methods for computing the autocorrelation coefficients were exponential in the number of inputs to the function(s). Since new methods for their computation have recently been introduced by Rice [8], [9], and by Stanković [10], I also have performed an investigation into development of software tools for calculating autocorrelation coefficients (spectrum) for Boolean function

In this paper I present the definition, explanation and software tools for calculating autocorrelation spectrum by using the Wiener-Kinchin theorem. The first tool calculates autocorrelation spectrum through matrix multiplication, the second tool use fast Walsh transform and the third tool calculates autocorrelation spectrum through decision diagrams. Then, I discussed "LGSynth93 benchmark" [11] testing space and time complexity statistics for each tool and tool's efficiency.

[1] Miloš M. Radmanović is with the Faculty of Electronics, University of Niš, Aleksandra lmedvedova 14, 18000 Niš, Serbia and Montenegro, e-mail: milos@elfak.ni.ac.yu

II. Autocorrelation of Switching Functions

The general cross-correlation (convolution) between two functions f and g at a distance τ is defined as:

$$\sum_{x=0}^{2^n-1} f(x)\cdot g(x\oplus\tau) \tag{1}$$

for two functions $f(X)$ and $g(X)$ where $X = x_n x_{n-1}\ldots x_1$. The symbol $\oplus$ represents the bitwise exclusive-or function and the symbol $\cdot$ represents arithmetic multiplication function. When $f = g$, the resulting equation gives the cross-correlation of the function with itself, translated by τ. The resulting coefficients are referred to as the autocorrelation coefficients of the function

The autocorrelation function is defined as follows:

$$B(\tau) = \sum_{x=0}^{2^n-1} f(x)\cdot f(x\oplus\tau) \tag{2}$$

where function is $f(X)$, $X = x_n x_{n-1}\ldots x_1$, $x = \sum_{i=1}^{n} x_i 2^{i-1}$, $\tau = \sum_{i=1}^{n} \tau_i 2^{i-1}$ and n is number of inputs [3].

For multiple output functions a second step must be performed to combine the autocorrelation function for each of the individual functions into the total autocorrelation function is defined as:

$$B(\tau) = \sum_{i=0}^{m-1} B_i(m) = \sum_{i=0}^{m-1}\sum_{x=0}^{2^n-1} f(x)\cdot f(x\oplus\tau) \tag{3}$$

where m is the number of outputs and the multiple output function F consists of $f_0 f_1 \ldots f_{m-1}$.

III. Wiener-Khinchin Theorem

The Wiener-Khinchin theorem states a relationship between the autocorrelation function and the Walsh (Fourier) coefficients [12].

$$B_f = 2^{-n} W^{-1}(Wf)^2 \tag{4}$$

where W denote the Walsh transform operator and W^{-1} denote the inverse Walsh transform operator.

An example of calculating the autocorrelation spectrum using Wiener-Kinchin theorem through matrix multiplication for the function $f(x_1, x_2) = x_1 + x_2$ is shown in the following equations:

$$B_f = 2^{-2} W(2)(W(2)F)^2 \tag{5}$$

$$B_f = \frac{1}{4}\begin{bmatrix}1 & 1 & 1 & 1\\ 1 & -1 & 1 & -1\\ 1 & 1 & -1 & -1\\ 1 & 1 & -1 & 1\end{bmatrix}\left(\begin{bmatrix}1 & 1 & 1 & 1\\ 1 & -1 & 1 & -1\\ 1 & 1 & -1 & -1\\ 1 & 1 & -1 & 1\end{bmatrix}\begin{bmatrix}0\\ 1\\ 1\\ 1\end{bmatrix}\right)^2 = \begin{bmatrix}3\\ 2\\ 2\\ 2\end{bmatrix} \tag{6}$$

Calculations in determination of the Walsh spectrum can be performed through the flow-graph describing the fast Walsh transform [13]. An example of calculating the autocorrelation spectrum using Wiener-Kinchin theorem through fast Walsh transform for the function $f(x_1, x_2) = x_1 + x_2$ is shown in the following equations:

$$\begin{matrix}0\\1\\1\\1\end{matrix} \xrightarrow{\substack{fast\\ Walsh\\ transform}} \begin{matrix}3\\-1\\1\\1\end{matrix} \xrightarrow{sqare} \begin{matrix}9\\1\\1\\1\end{matrix} \xrightarrow{\substack{fast\\ Walsh\\ transform}} \begin{matrix}12\\8\\8\\8\end{matrix} \xrightarrow{1/4} \begin{matrix}3\\2\\2\\2\end{matrix} \tag{7}$$

Calculations in determination of the Walsh spectrum can be performed through fast Walsh transform over decision diagram [10]. An example of calculating the autocorrelation spectrum using Wiener-Kinchin theorem through fast Walsh transform and decision diagrams for the function $f(x_1, x_2) = x_1 + x_2$ is shown in the following figure:

IV. SOFTWARE TOOLS

My applications (software tools) are written in Microsoft Visual Studio [14] and use MFC technology.

The first software tool for calculating the autocorrelation spectrum using Wiener-Kinchin theorem through matrix multiplication uses dynamic arrays of arrays as data structure for storing true vector and Walsh transform matrix. Because autocorrelation coefficient can be number between 0 and $2^n - 1$, it is used 64-bit integer data type for storing autocorrelation spectrum. Using calculating the autocorrelation spectrum by Wiener-Kinchin theorem for a function requires $O(2^{2n})$ operations (multiplication and summing) on dynamic arrays, where n is number of inputs Eq. (5) and (6).

Using another data structures and methods requires significantly less operations for calculating the autocorrelation spectrum

The second software tool for calculating the autocorrelation spectrum using Wiener-Kinchin theorem through fast Walsh transform over vectors uses dynamic arrays as data structure for storing true vector. This tool uses much less memory then first software tool. The tool uses only one 64-bit integer array and it don't produce any temporary arrays. Using calculating the autocorrelation spectrum by Kinchin theorem for a function requires $O(n2^n)$ operations (multiplication and summing) on dynamic arrays, where n is number of inputs Eq. (7). Using decision diagrams as data structure and DD method requires significantly less operations in some cases of calculating the autocorrelation spectrum.

The third software tool for calculating the autocorrelation spectrum using Wiener-Kinchin theorem through fast Walsh transform over decision diagrams uses special case of linked lists as data structure. The tool was develop following basic programming principles for DD packages [15]. Every DD package use an imperative programming language like C++, nodes are class structures that contain a variable identifier and "then" and "else" children pointers; a "next" pointer strings nodes together that belong to the same collision chain in the unique table, recycling of nodes is easily implemented by keeping a reference count for each node. Using calculating the autocorrelation spectrum by Kinchin theorem for a function requires $O(i) + O(j) + O(k) + O(l)$ operations (multiplication and summing linked lists) on special case of linked lists, where i is number of nodes of initial DD, l is number of nodes of temporary DD (after first Walsh transform), k is number of nodes of temporary DD (after squaring) and l is number of nodes of resulting DD (after second Walsh transform), (see figure 1).

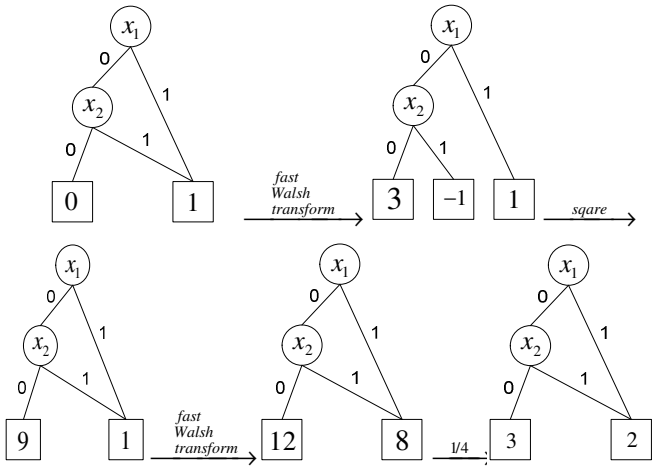

Fig. 1. An example of calculating the autocorrelation spectrum using Wiener-Kinchin theorem through fast Walsh transform and decision diagrams

V. TESTS AND RESULTS

Below I give a lists and tables of different tools testing statistics. I performed the testing on a PC Pentium IV on 1,4 GHz with 224 MB of RAM (MS Windows XP Professional 2002). The memory usage for all tools was limited to 150 MB, and timing statistics do not include building and storing data structures for functions (arrays of arrays, arrays and linked lists). I have tasted tools with "LGSynth93 benchmark", and I

used input files described in ESPRESSO-MV (or pla) format [16].

LGSynth93 benchmark suites is based on the collection of benchmarks from the ISCAS85, ISCAS86 and LGSynth91. testing space and time complexity statistics for each tool and tool's efficiency. The benchmarks then have been categorized in three categories: small benchmarks, medium sized benchmarks (and too-large benchmarks. This allows to judge the quality of tools and gives a better overview of the existing benchmarks. Descriptions and properties of switching functions from LGSynth benchmark suite are given in tables 1, 2 and 3.

Table 4 describes tool efficiency statistics and it shows that most efficient tool is third tool that use fast Walsh transform through array. But, for too-large benchmarks, neither tool is efficient (25% isn't enough).

TABLE I

LGSYNTH93 SMALL BENCHMARKS

Name	Number of inputs	Number. of outputs	Number of cubes
xor5	5	1	16
rd53	5	3	32
squar5	5	8	32
bw	5	28	87
con1	7	2	9
rd73	7	3	141
inc	7	9	34
5xp1	7	10	75
sqrt8	8	4	40
rd84	8	4	256
misex1	8	7	32
9sym	9	1	87
clip	9	5	167
apex4	9	19	438
sao2	10	4	58
ex1010	10	10	1024
alu4	14	8	1028
table3	14	14	175
misex3c	14	14	305
misex3	14	14	1848
b12	15	9	431
t481	16	1	481
pdc	16	40	2810
spla	16	46	2307
table5	17	15	158

Table 5 describes average time over all benchmarks and it shows that best average time for small and medium sized

TABLE II

LGSYNTH93 MEDIUM SIZED BENCHMARKS

Name	Number of inputs	Number. of outputs	Number of cubes
duke2	22	29	87
cordic	23	2	1206
cps	24	109	654
vg2	25	8	110
misex2	25	18	29

TABLE III

LGSYNTH93 TOO-LARGE BENCHMARKS

Name	Number of inputs	Number. of outputs	Number of cubes
apex2	39	3	1035
seq	41	35	1459
apex1	45	45	206
apex3	54	50	280
e64	65	65	65
apex5	117	88	1227
ex4p	128	28	620
o64	130	1	65

TABLE IV

TOOL EFFICIENCY STATISTICS

Tool	Small benchmarks	Medium sized benchmarks	Too-Large banchmarks
first. tool	87%	0%	0%
second tool	100%	60%	0%
third tool	100%	100%	25%

TABLE V

AVERAGE TIME STATISTICS

Tool	Small benchmarks	Medium sized benchmarks	Too-Large banchmarks
first. tool	46.863	-	-
second tool	1.667	740.797	-
third tool	1.273	0.741	66.561

TABLE VI

MAX SPACE STATISTICS (ELEMENTS OF ARRAY)

Tool	Small benchmarks	Medium sized benchmarks	Too-Large banchmarks
first. tool	~2^34	~2^50	~2^130
second tool	~2^17	~2^25	~2^65

benchmark has third tool. It is expected, because the calculation time is direct proportional to tool's space (memory) request.

Table 6 describes tool maximal space statistics (number of elements in arrays) and it shows that first tool for medium sized benchmark require almost 2^50*32 bit = 4096 TB and for too-large benchmark 268435456 TB of memory. It is expected; because the space is exponential proportional to function's number of inputs.

Table 7 describes third tool space statistics (number of decision diagram nodes) and shows that we can not calculate space limit and dependency. In most cases, it is shown that $i \sim j \sim k \sim l$, but functions like *cps* ond *apex5* shows that $i << j$. Functions: *alu4*, *table3*, *misex3c* and *misex3* shows that $i << l$. Meanwhile, all functions shown that $j \sim k$. If it is possible. to calculate Walsh spectrum over DD, there is high probability for calculation of autocorrelation spectrum.

TABLE VII

SPACE STATISTICS (NUMBER OF DD NODES) FOR THIRD TOOL

Name	**i** (initial DD.)	**j** (after 1. Walsh T..)	**k** (after . squaring.)	**l** (after 2. Walsh T.)
xor5	9	5	5	9
rd53	23	34	29	29
squar5	38	83	56	58
bw	114	330	236	253
con1	18	76	50	30
rd73	43	57	44	51
inc	89	371	280	209
5xp1	88	297	208	237
sqrt8	42	123	75	126
rd84	59	88	48	55
misex1	47	276	159	88
9sym	33	39	39	37
clip	254	529	326	539
apex4	1021	4836	3979	4595
sao2	154	481	394	472
ex1010	1079	6281	5292	5696
alu4	1352	6195	4351	15364
table3	941	41652	33660	40295
misex3c	847	9189	6851	16077
misex3	1301	18364	13009	55520
b12	91	651	492	151
t481	32	184	60	60
pdc	705	10717	6457	3826
spla	681	8785	5480	3545
table5	873	89589	75130	48959
duke2	976	6581	4590	5273
cordic	80	646	491	463
cps	2318	15420	8292	5339
vg2	1059	3793	3096	6563
misex2	140	1311	844	139
apex2	7102	-	-	-
seq	142321	-	-	-
apex1	28414	-	-	-
apex3	-	-	-	-
e64	1446	6853	5473	2144
apex5	2705	74258	31269	4634
ex4p	1301	-	-	-
o64	-	-	-	-

VI. CONCLUSION

In this paper I present software tools for calculating autocorrelation spectrum by using the Wiener-Kinchin theorem. The first tool calculates autocorrelation spectrum through matrix multiplication, the second tool use fast Walsh transform and the third tool calculates autocorrelation spectrum through decision diagrams. Then, I presented "LGSynth93 benchmark" testing statistics (efficiency, time, space) for each tool. Third DD-based tool has best results over all benchmarks, but for too-large benchmarks, tool require further work to optimize in terms of memory.

ACKNOWLEDGMENT

This research work was based on a scholarship of The German Academic Exchange Service (DAAD) under the Stability Pact for South-East Europe.

REFERENCES

[1] Tomczuk, R., "Autocorrelation and Decomposition Methods in Combinational Logic Design", 1996, PhD dissertation.

[2] Rice, J.E., Serra, M., Muzio, J.C., "The Use of Autocorrelation Coefficient for Variable Ordering for ROBDDs", Proc. 4th Int. Workshop on Applications of Reed-Muller Expansion in Circuit Design, Victoria, Canada, August 20-21, 1999, 185-196.

[3] Karpovsky M.G., "Finite Orthogonal Series in the Design of Digital Devices", John Wiley 1976.

[4] Ruce, J.E., Muzio, J.C., "The Use of Autocorrelation Function in Classification of Switching Functions", In. Euromicro Symposium on Digital System Design: Architectures, Methods and Tools (DSD), pages 244-251, 2002.

[5] Ruce, J.E., Muzio, J.C., "On The Use of Autocorrelation Coefficients in The Identification of Three-level Decompositions", Proc. Int. Workshop on Logic Synthesis, (IWLS 2003), 2003.

[6] S. L. Hurst, D. M. Miller, and J. C. Muzio, "Spectral Techniques in Digital Logic", Orlando, Florida: Academic Press, Inc., 1985.

[7] Rice, J.E., Muzio, J.C., "Properties of Autocorrelation Coefficients", Proc. IEEE Pacific Rim Conf. on Communications, Computers and Signal Processing, 2003.

[8] Rice, J.E., Muzio, J.C., "Methods for Calculating Autocorrelation Coefficients", In Proceedings of the 4th Workshop on Boolean problems, pages 69-76, 2000.

[9] Rice, J.E. "Autocorrelation Coefficients in the Represen-tation and Classification of Switching Functions", PhD thesis, University of Victoria, 2003.

[10] Stanković, R., S., Bhattacharaya, M., Astola, J., T., "Calulation of Dyadic Autocorrelation through Decision Diagrams", Proc. European Conf. Circuit Theory and Design, pp. 337-340, 2001.

[11] "The LGSynth93 Benchmark Suit", Available: http://www.bdd-portal.org/benchmarks/LGSynth93.tar.gz.

[12] Pichler F., "Walsh Functions and Linear System Theory", Proc. Applic. Walsh Functions, Washington, D.C., 1970, 175-182.

[13] Clarke, E., M., McMillan, K.L., Zhao, X., Fujita, M., "Spectral Transforms for Extremely Large Boolean Functions in Kebaschull", U., Schubert, E., Rosenstiel, W., Eds., Proc. IFIP WG 10.5 Workshop on App. of the RM Expression in circuit Design, Humburg, Germany, September 16-17, 1993, 86-90.

[14] Microsoft Corporation, Available: www.microsoft.com

[15] Jansen G. "Design of Pointerless BDD Package" inite", 10th Worshop on Logic and Synthesis Granlibakken, Lake Tahoe, CA, 12-15, 2001.

[16] Lisanke, R., "Logic Synthesis and Optimization Benchmarks User Guide" Version 2.0, 1988.