Journal of Achievements in Materials and Manufacturing Engineering
VOLUME 57
ISSUE 2
April
2013

The evaluation of the TPM synchronization on the basis of their outputs

M. Dolecki [a,*], R. Kozera [a,b], K. Lenik [c]
[a] Faculty of Mathematics, IT and Landscape Architecture,
John Paul II Catholic University of Lublin,
ul. Konstantynów 1h, 20-708 Lublin, Poland
[b] Faculty of Applied Informatics and Mathematics,
Warsaw University of Life Sciences - SGGW,
ul. Nowoursynowska 159, 02-776 Warszawa, Poland
[c] Department of Fundamental Technics, Lublin University of Technology,
ul. Nadbystrzycka 38, 20-618 Lublin, Poland
* Corresponding e-mail address: michal.dolecki@kul.pl

Received 29.01.2013; published in revised form 01.04.2013

Analysis and modelling

ABSTRACT

Purpose: Tree Parity Machines are specific artificial neural networks used to construct relatively secure key exchange protocol [12,15,24]. The level of networks' compatibility is measured by weight vectors mutual overlap. However, to calculate such mutual overlap, one needs to be familiar with both weights' vectors, which is impossible in practical key exchange. This paper discusses other schemes to evaluate compatibility of weights' vectors. The first one uses Euclidean distance of both weights' vectors. The second one is based on frequencies of common TPM's outputs and as such does not rely on the weights' vectors. Both approaches to handle secure key exchange protocol facilitate more extended analysis of many technical processes in which a vital role plays an incorporation of a non-standard high-quality method securing any sensitive data.
Design/methodology/approach: Computer simulations of TPM synchronization are conducted using authors' program and the obtained results are statistically analyzed herein.
Findings: We found experimentally that mutual overlap of the weights' vectors is highly correlated with Euclidean distance. Additionally, frequencies of common outputs in given numbers of learning cycles stay in high correlation with this mutual overlap and Euclidean distance. The latter can subsequently be used to draw pertinent conclusions about TPM's weights compatibility.
Practical implications: Proposed methods, especially frequencies analysis, can be applied to key exchange protocol to improve its security. Determining the vectors compatibility level before synchronization completion allows qualifying this synchronization to one of the possible time classes.
Originality/value: New ideas presented in this work involve application of Euclidean distance and common output frequencies to calculate the networks compatibility given by weights mutual overlap.
Keywords: Artificial Intelligence Methods; Neurocryptography; Data security

Reference to this paper should be given in the following way:
M. Dolecki, R. Kozera, K. Lenik, The evaluation of the TPM synchronization on the basis of their outputs, Journal of Achievements in Materials and Manufacturing Engineering 57/2 (2013) 91-98.

1. Introduction

The main purpose of cryptography is to provide a safe method of communication between two or more persons (or among the other different entities). Substantial amount of communication takes place on an open channel. Therefore there are always risks that sent message will be intercepted by unauthorized parties eager to infiltrate its content [9]. Many sophisticated algorithms are developed to ensure, with the satisfactory level of certainty the preservation of communication confidentiality. The sender, using these algorithms, transforms the plaintext into the cryptogram and sends it to the recipient, which in turn is able to transform it back to the original message. Encryption and decryption algorithms perform transformations that depend on additional data called cryptographic keys. Thanks to the additional keys cryptosystem, security is shifted from the used algorithms to the problem of distribution and the security of the used keys. In asymmetric cryptography, the sender and receiver use a pair of keys, one of which is used to encrypt and the second one to decrypt messages [19]. In symmetric cryptography, the sender and receiver use the same key for encryption and decryption [19]. A major problem thus becomes a key management, in particular to ensure the safety of their distribution. This problem was solved in 1976 by Diffie and Hellman [19,26], who published the idea a key exchange protocol with the use of an open channel. It is based on computationally hard problem which is the calculation of the discrete logarithm problem in the cyclic group. The use of this type of problems for construction of cryptosystems guarantees the fulfillment of crucial security inequalities, which means that the cost of breaking security must be disproportionate to the benefits of breaking the cipher.

An interesting alternative for cryptosystems based on number theory can be neurocryptography [13,14]. Artificial neural networks are effective and widely used analytic tools. They are also applied, as other artificial intelligences' methods, to research in material science [1-5, 16,17] or in e-foresight [6,7]. Characteristic for these networks is ability to learn from examples, which allows solving given problem without constructing a classical algorithm [6,25]. Neurocryptography introduces artificial neural networks as a tool for encryption and decryption. It also serves as a cryptographic key exchange method on an open channel. The latter approach is based on the discovered by Kanter et. al. [11,12] the phenomenon of the networks' synchronization during their mutual learning. At the beginning of the key exchange procedure, each partner generates random values of their networks' weights and this initial state is kept in secret. During mutual learning both networks' weight vectors generally coincide to establish common values, which can be used as a cryptographic key in further communication. For this particular application a specific multi-layer, feed-forward network with a tree-like structure is used. Such tree with this special topology is commonly called a Tree Parity Machine (in abbreviation TPM).

Network synchronization process is a stochastic process rendering networks' weights modification according to certain algorithms of learning. The most important parameter describing the dynamics of this process is the value of weights' vectors mutual overlap [23]. In order to determine such overlap, one needs to know the weights' vectors of both synchronized networks. However, the practical application of this method, requires that both weights' vectors remain secret. The method presented here relies on measuring the frequencies of equal networks output in a given number of previous steps. We found that these frequencies are in significant linear relationship with the mutual overlap. We also propose here an alternative method of evaluating the level of synchronization of the network based on the Euclidean distance between both weights' vectors. Our tests indicate a strong correlation between Euclidean distance and the currently used mutual overlap. In addition, the normalized distance between vectors is more strongly correlated with the measured frequencies of TPM's equal outputs than with the mutual overlap. This research permits to evaluate the level of network synchronization without knowing their weights' vectors.

Tree Parity Machine network is a multi-layered, feed-forward network with disjoint receptive fields [12]. Diagram of such network is presented in Fig. 1. The illustrated structure can be defined by three parameters: K-N-L. Here integer K determines the number of artificial neurons in the hidden layer. The number N yields, in turn the quantity of input values for each neuron. The admitted output fired by each neuron is assumed to be either 1 or -1 according to following formula:

$$\sigma_j = \begin{cases} -1, if \ \sum_{i=1}^{N} w_{ji} x_{ji} \leq 0, \\ 1, if \ \sum_{i=1}^{N} w_{ji} x_{ji} > 0. \end{cases} \tag{1}$$

The signal reaching the neuron is multiplied by a corresponding weight, as was shown in sums above. Each weight w_{ji} is an integer falling within the range of -L to L. Each TPM network has KN of such weights, so it can take one of the $(2L+1)^{KN}$ states. At the beginning of synchronization process both partners choose at random, different values of weights for their network. The initial state of each network is kept secret, and because of the use of public input values, the disclosure of the initial weights values would allow to break this protocol. For two networks the number of possible states increases to the $(2L+1)2^{KN}$ Output of the whole network is calculated as the product of the outputs of the hidden layer neurons which has a value of either 1 or -1, according to the formula:

$$\tau = \prod_{j=1}^{K} \sigma_j \tag{2}$$

Each of the output values can be obtained as one of the 2K-1 combinations of the results of the hidden layer. For example, for a network with K=3 and output equals to 1, there could be one of four combinations of the hidden layer results D={(1,1,1), (-1,-1,1), (-1,1, -1) , (1, -1, -1)}.

Hebbian rule [10,20,22] used for network learning, modifies the value of weights if the result of the network is consistent with the expected result. In the TPM network synchronization, the role a teacher is played by other party's network. If two networks return the same outcomes for the common input vector, the weight of these neurons, which had result consistent with the result of the entire network are modified to strengthen their connection. Returning to our example from above, if the networks output were 1 and hidden layer neurons had first combination

from the set D, the weights of all hidden neurons would be changed. For other combinations of outputs only one neuron's weights is modified (the third, the second and the first one, respectively). The attacker does not know the internal representation of the result of the attacked network as, at each step synchronization, involving a change of weight, he has 2^{K-1} possible options of weight changes. To synchronize the network one of three methods of learning can be invoked:

- Anti-Hebbian rule [11-15]. Weights' modification complies here with the following formula:

$$w_{ki}^{(t+1)} = w_{ki}^{(t)} - x_{ki}\sigma_k. \tag{3}$$

- Hebbian rule [15,23]. Modification is defined according to the rule:

$$w_{ki}^{(t+1)} = w_{ki}^{(t)} + x_{ki}\sigma_k \tag{4}$$

- Random Walk rule [23]. Modification is given by the formula listed below:

$$w_{ki}^{(t+1)} = w_{ki}^{(t)} + x_{ki}. \tag{5}$$

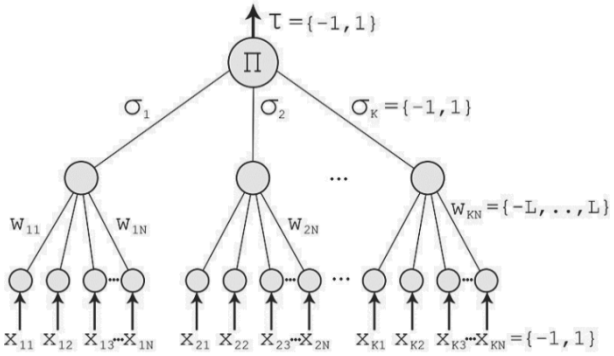

Fig. 1. Tree Parity Machine topology

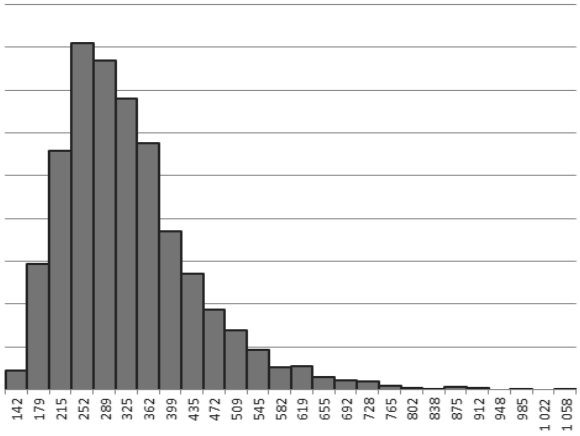

Fig. 2. Synchronization time histogram for TPM 3-101-3

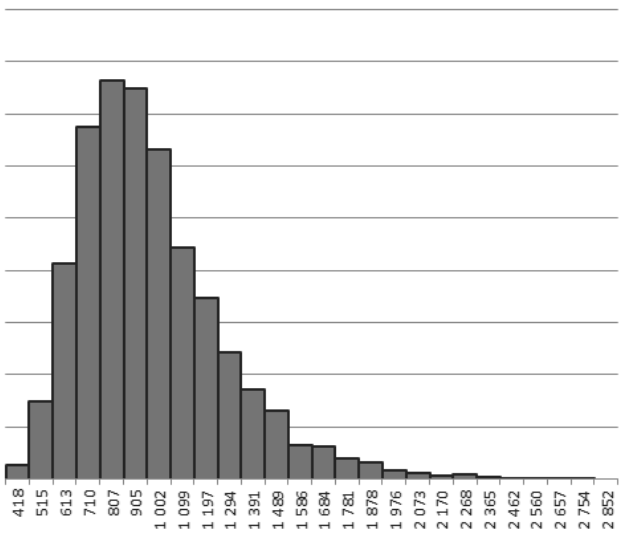

Fig. 3. Synchronization time histogram for TPM 3-101-5

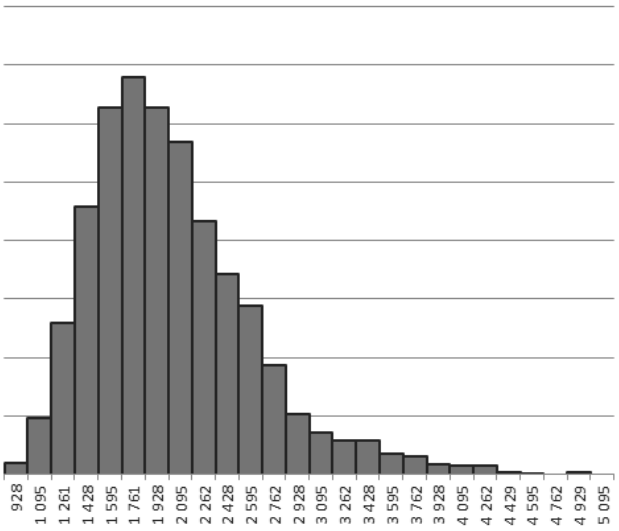

Fig. 4. Synchronization time histogram for TPM 3-101-7

In the first rule weights are modified if TPM's outputs are different and whole process leads to reverse weights' vectors. Other methods change weights when outputs are equals and give the same weights' vectors in both networks. Once both networks reach the state of synchronization they stay in such synchronized mode regardless of the time devoted for further learning. In addition, since the weights vectors are compatible, the networks return the same results for the common input vector. Long enough exchange of networks' compatible results indicates their synchronization to be completed.

Synchronization of the examined TPM networks is astochastic process, and the time required to achieve full synchronization forms the left-sided histogram. We present now the pertinent synchronization time histograms for the Tree Parity Machine

networks with the respective topologies: 3-101-3, 3-101-5 and 3-101-7 learned using Random Walk rule. These histograms are created upon analysis of 5000 synchronizations for each network (see Figs. 2-4). The corresponding generated results are subsequently divided into 26 classes. Despite the fact that these networks synchronize upon different elapsing times, their respective histograms indicate a similar behaviour pattern. In fact their shapes show a big degree of similarity.

As justified in [8] up to 75% of networks synchronize in less than half of the longest observed synchronization time. Shorter synchronization time improves system safety by reducing the amount of information available to an attacker, and the time to teach his network.

2. Results and discussion

Analysis of TPM synchronization process conducted for this method is based on the knowledge of both network weights' vectors. The main parameter determining the level of network synchronization is weights' vectors mutual overlap calculated for each hidden layer neuron using the formula (here ○ denotes the dot product):

$$\rho_i^{AB} = \frac{w_i^A \circ w_i^B}{\sqrt{w_i^A \circ w_i^A} \cdot \sqrt{w_i^B \circ w_i^B}}, \tag{6}$$

where i is the index of neuron in network, and w_i^A and w_i^B are weight vectors of i-th neuron of the networks A and B, respectively. More precisely, the calculated parameter is the cosine of the angle between the weights vectors of i-th hidden layer neuron. At the beginning of the synchronization, ρ_i^{AB} has a value close to 0, and subsequently along the iteration process is being changed to achieve a value 1 for a synchronized network. For a fully synchronized network weights' vectors overlap equals 1 because the angle between them is equal to 0. Further analysis is based on the connected weights' vectors of all hidden layer neurons. It consists of KN integers ranging from -L to L. Knowing both partner's A and B network weights' vectors, one can measure a total mutual overlap over entire network analogously to the measurements of mutual overlap for each single neuron:

$$\rho_{\square}^{AB} = \cos\theta = \frac{w_{\square}^A \circ w_{\square}^B}{\sqrt{w_{\square}^A \circ w_{\square}^A}\sqrt{w_{\square}^B \circ w_{\square}^B}} = \frac{w_{\square}^A \circ w_{\square}^B}{\|w_{\square}^A\| \cdot \|w_{\square}^B\|} \tag{7}$$

Another scheme to determine the compatibility of weights' vectors is to calculate the distance between them using the Euclidean metric. Such distance between two vectors of weights for both networks is given by the well-known formula:

$$dist(A,B) = \|w^A - w^B\| = \sqrt{\sum_{k=1}^{KN}(w_k^A - w_k^B)^2}. \tag{8}$$

In this case, the index k indicates the value of the specific weight from the networks' connected weight vectors. Relationship of the cosine of the angle between the weights' vectors and the distance between them is given by the cosine theorem holding for arbitrary unitary spaces:

$$\|w^A - w^B\|^2 = \|w_{\square}^A\|^2 + \|w_{\square}^B\|^2 - 2 \cdot \|w_{\square}^A\| \cdot \|w_{\square}^B\| \cdot \cos\theta. \tag{9}$$

An important difference between the cosine of the angle between two vectors, and the distance between them is the range, which includes the value of both functions and changes these values during synchronization of the networks. Cosines of the initial weights' vectors take values close to 0 and keep growing to ultimately approach the value 1 for fully synchronized networks. Euclidean distance at the beginning of the synchronization takes large values depending on the size of the network and the range, which includes the weight, while for the synchronized network such distance is set to 0.

As an example we consider the two TPM networks with the selected topology of the type 3-101-3. The first one synchronizes within 348 cycles. This number coincides with the average synchronization cycle number computed for the nets of the analogous size and simulated within 5000 synchronizations. The second net synchronizes in 620 cycles and this time belongs to the middle class of histogram presented above. Fig. 5 presents cosine and Euclidean distance values for first TPM, and Fig. 6 the same values for the second TPM. Given a large disproportion occurred within the ranges between the corresponding cosine values and Euclidean values Figs. 5 and 6 contain two ordinates i.e. two Y-axes.

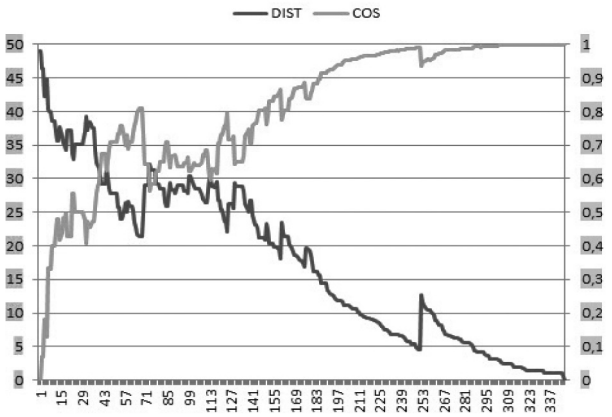

Fig. 5. Euclidean distance and cosine of the first TPM

For the comparison between these two values it is therefore necessary to rescale them to a common set of values. For this purpose, a reversal and normalization distance is applied according to the formula:

$$dist(A,B)_t = 1 - \frac{dist(A,B)_t - \min_{1 \le j \le t_{synch}} dist(A,B)_j}{\max_{1 \le j \le t_{synch}} dist(A,B)_j - \min_{1 \le j \le t_{synch}} dist(A,B)_j}. \tag{10}$$

Fig. 7 and Fig. 8 show comparison of cosine and normalized and reversed Euclidean distance for the same synchronization as it is shown earlier in Fig. 5 and Fig. 6.

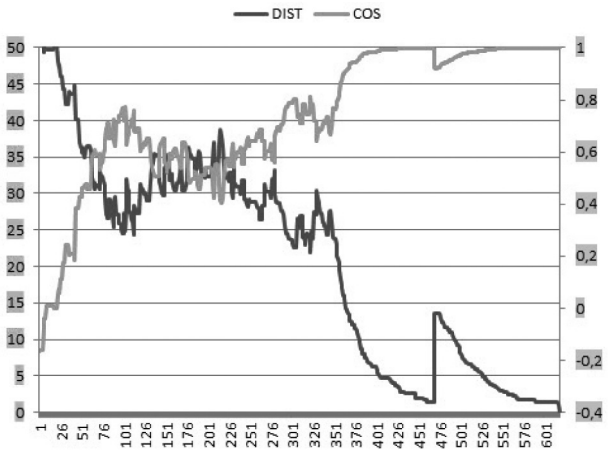

Fig. 6. Euclidean distance and cosine of the second TPM

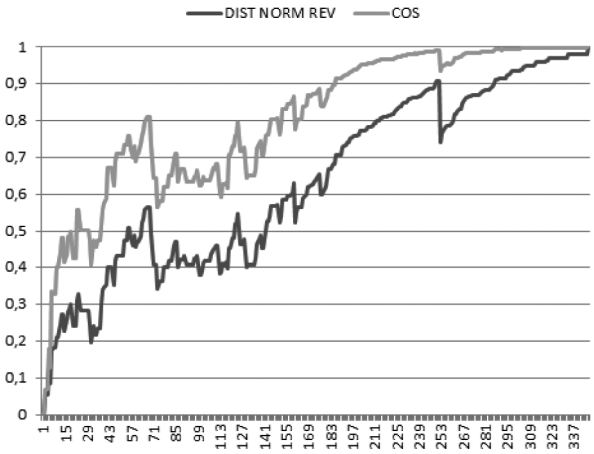

Fig. 7. Normalized and Reversed Euclidean distance and cosine for the first TPM

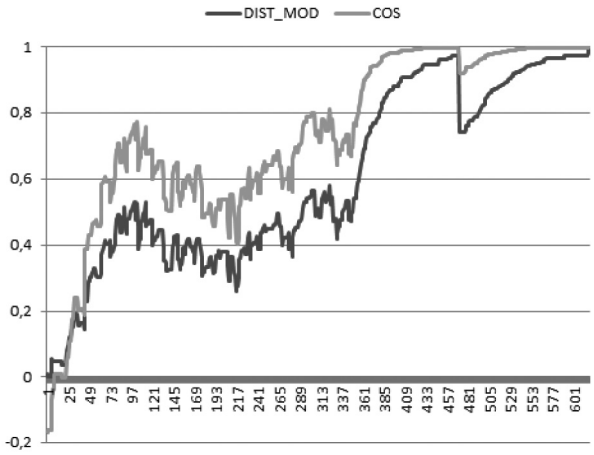

Fig. 8. Normalized and Reversed Euclidean distance and cosine for the second TPM

It is visible that both values are strongly correlated.

The obtained distances for analyzed networks are highly correlated with the respective values of cosine. As demonstrated by experimental means, the correlation coefficient of mutual overlap and Euclidean distance for the network in Table 1 exceeds 0.95. This means that the proposed method for evaluating the degree of synchronization is as effective as the methods used so far. In addition normalized distances tend to be more correlated with the frequency occurrence of the network consistent results.

The practical application of examined phenomenon appearing during TPM network synchronization for the construction key exchange protocol requires that the network weights are kept in secret. Communication parties cannot therefore simply use the methods described above. The only knowledge for partners exchanging key is the result of the network other part for the common input vector. Based on the available results of both networks, partners can set the frequency with which both networks have the same results. Using these frequencies, one can reason on a compatibility of network weights' vectors. The method presented below allows the calculation of the frequency of equal networks results in *s* previous steps, where *s* is a given natural number. Created sequence illustrates the dynamics of the network synchronization.

Let the (a_n) be a sequence, and $a_n=1$ if $\tau_n^A = \tau_n^B$ and $a_n=0$ if $\tau_n^A \neq \tau_n^B$. Index *n* indicates the number of learning step and $n=1,2,\ldots,t_{synch}$. Let sequence (b_n) be defined as an average of *s* elements from (a_n) with indices $n,n-1,\ldots,n-s+1$. If index *n* is less than number of elements to analyze *s*, then in sequence (a_n) there aren't *s* elements before *n*-th element, and it has to be calculated as an average of all available elements form the first one. This sequence is given by formula:

$$b_n = \frac{1}{l}\sum_{j=n-l+1}^{n} a_n, \tag{11}$$

where l=min(n,s).

The analysis conducted herein indicates, that for the majority of the analyzed networks, a significant linear relationships hold between the frequencies of common TPM's outputs and the mutual overlap, as well as between these frequencies and Euclidean distance. However, the second correlation is usually stronger. For networks synchronizing in a longer time the correlation is little lower and suggests a moderate correlation. This means, that the communication partners can infer about the compatibility of weights' vectors by analyzing the frequency of consistent results, without knowing the other parties weights' vector.

Figs. 9 and 10 show normalized and reversed Euclidean distance and frequencies for s=75 and s=125. First diagram shows frequencies divided by l, as presented in formula above. The sum of consistent results for both nets, under fixed preceding steps is divided by the quantity of accessible results. The latter does not yield satisfactory adjustment fitting at the beginning of the synchronization process.

A simple inspection shows, that although the first parts of the generated plots above have different values, they can still be much better adjusted upon dividing computed frequencies by s instead of dividing them by l. Once the division by 1 is applied, the number of consistent results of both networks obtained in the previous steps is divided by 1. It can turn out that the occurrence of the few consistent results of both networks at the initial phase of synchronization may yield high frequencies thanks to the small

value of the divisor 1. The latter implies, that despite insignificant consistency of weights' vectors the computed frequency of consistent results is still high. To handle this inconvenience, so that the frequencies' values are better fitted to the consistency of the respective weights' vectors the above mentioned divisor 1 is increased to s. The number s is determined on the basis of the quantity of the so-far analyzed steps. For the initial synchronization cycles we observed the cases when the number of all steps is smaller than s. In such a case the analysis of all 1 cycles with 1<s is performed and consequently in such eventuality the quantity of consistent results is divided by 1. A slight modification of this approach is also here accomplished. Namely, irrespectively from the number of accessible cycles the quantity of consistent results is always divided by an a priori specified length s of the cycles. This procedure is still conducted if the corresponding summation runs over only one index. The respective frequencies derived with this scheme are much better fitted to the real distance between the weight's vectors.

The alternative sequence is given by formula:

$$b_n = \frac{1}{s}\sum_{j=n-l+1}^{n} a_n, \tag{12}$$

where l=min(n,s).

Fig. 11 and Fig. 12 shows these frequencies and Euclidean distance.

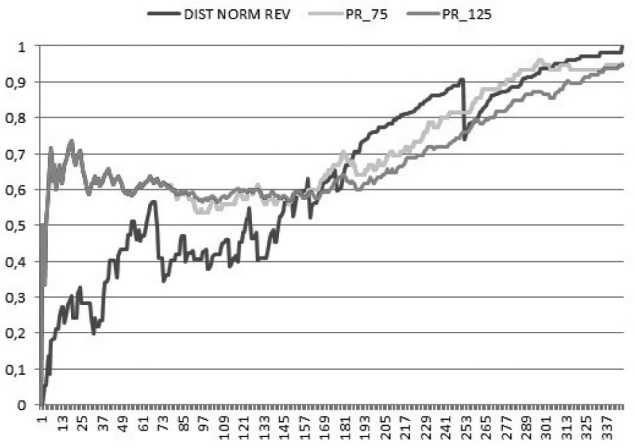

Fig. 9. Normalized and Reversed Euclidean distance and frequencies divided by l for the first TPM

Table 1.
Results of analysis

Synchronization time		Correlation dist and cos	FR 25	FR 75	FR 125	FR 175	FR 225
150		0.952					
	corr dist fr		0.933	0.948	0.919	0.913	0.913
	corr cos fr		0.943	0.930	0.937	0.946	0.946
200		0.970					
	corr dist fr		0.890	0.822	0.755	0.747	0.747
	corr cos fr		0.795	0.680	0.613	0.618	0.621
250		0.962					
	corr dist fr		0.891	0.934	0.934	0.934	0.935
	corr cos fr		0.813	0.853	0.858	0.864	0.890
300		0.958					
	corr dist fr		0.841	0.833	0.801	0.782	0.791
	corr cos fr		0.712	0.684	0.651	0.641	0.660
350		0.967					
	corr dist fr		0.808	0.831	0.771	0.698	0.617
	corr cos fr		0.700	0.688	0.609	0.521	0.432
400		0.952					
	corr dist fr		0.737	0.783	0.739	0.754	0.750
	corr cos fr		0.593	0.637	0.628	0.675	0.661
450		0.963					
	corr dist fr		0.701	0.656	0.555	0.463	0.368
	corr cos fr		0.519	0.446	0.327	0.224	0.123
500		0.968					
	corr dist fr		0.692	0.702	0.679	0.645	0.623
	corr cos fr		0.579	0.566	0.532	0.493	0.473

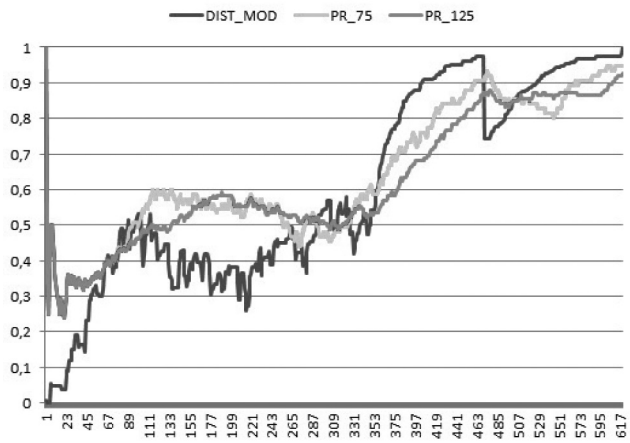

Fig. 10. Normalized and Reversed Euclidean distance and frequencies divided by l for the second TPM

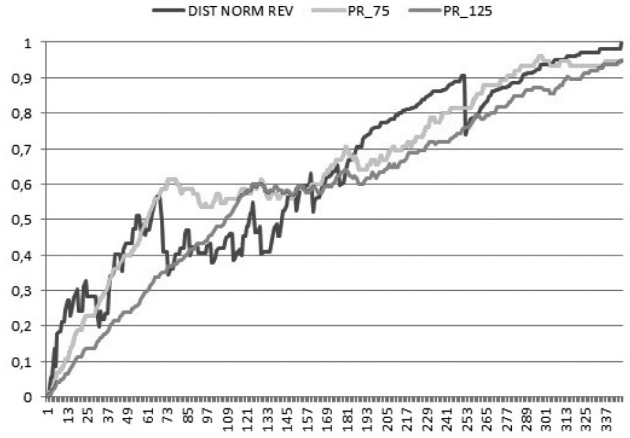

Fig. 11. Normalized and Reversed Euclidean distance and frequencies divided by s for the first TPM

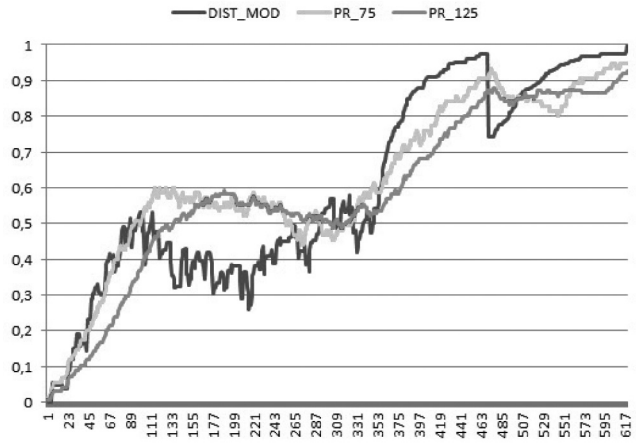

Fig. 12. Normalized and Reversed Euclidean distance and frequencies divided by s for the second TPM

Clearly, the computed herein frequencies of the consistent exchange results for both nets correspond very strongly to the weights' distance.

In our testing, 5000 synchronizations are carried out for the network with the structure 3-101-3. The respective synchronization times are within the range from 142 to 1095 cycles with the average 348 cycles. Table 1 contains a summary of correlation coefficients for the analyzed network. It illustrates networks with synchronization time between 150 and 500 cycles and the step 50 cycles. For each network it specifies a) the correlations between distance and the cosine of weight vector, b) correlation coefficients of sampled frequencies from 25 to 225 with step 50 and the Euclidean distance and c) correlation coefficients this frequency and cosine, respectively.

The new approach proposed in this paper may serve as a non-standard and very safe tool to secure sensitive input/output technical data. In particular this is important for the functioning of new unique and innovative technologies (see e.g. [18] or [21]).

3. Conclusions

Histogram describing the frequencies of network synchronization in a specified number of learning cycles indicates that usually networks synchronize relatively quickly. There are, unfortunately, probabilities for which the network will need to synchronize with the large number of cycles well above the experimentally computed average synchronization time**.** In the worst analyzed case, long synchronization takes over 2.7 times more cycles then the respective average synchronization. Without knowing the weights' vector it would be difficult to categorize whether the current synchronization falls in the fast group or whether it will last longer. Our research indicates, that the frequencies of equal TMP's output are strongly correlated with the weights' vectors mutual overlap and with the vectors computed with the aid of Euclidean distance. Thus, the analysis of the frequencies presented here can be exploited by both communication partners, to assess the networks synchronization level. Consequently, each investigated synchronization process in question can be classified to either the long or the short one (or alternatively can be qualified into one of the predefined time duration zones).

References

[1] L.A. Dobrzański, R. Honysz, Informative technologies in the material products designing, Archives of Materials Science and Engineering 55/1 (2012) 37-44.

[2] L.A. Dobrzański, R. Honysz, Application of artificial neural networks in modelling of quenched and tempered structural steels mechanical properties, Journal of Achievements in Materials and Manufacturing Engineering 40/1 (2010) 50-57.

[3] L.A. Dobrzański, R. Maniara, J.H. Sokolowski, W. Kasprzak, M. Krupiński, Z. Brytan, Applications of the artifcial intelligence methods for modeling of the ACAlSi7Cu alloy crystallization process, Journal of Materials Processing Technology 192-193 (2007) 582-587.

[4] L.A. Dobrzański, M. Staszuk, R. Honysz, Application of artificial intelligence methods in PVD and CVD coatings properties modelling, Archives of Materials Science and Engineering 58/2 (2012) 152-157.
[5] L.A. Dobrzański, M. Staszuk, R. Honysz, Application of artificial neural networks in properties modelling of PVD and CVD coatings, Archives of Computational Materials Science and Surface Engineering 2/3 (2010) 141-148.
[6] A. Dobrzańska-Danikiewicz, E-foresight of materials surface engineering, Archives of Materials Science and Engineering 44/1 (2010) 43-50.
[7] A. Dobrzańska-Danikiewicz, L.A. Dobrzański, J. Mazurkiewicz, B. Tomiczek, Ł. Reimann, E-transfer of materials surface engineering e-foresight results, Archives of Materials Science and Engineering 52/2 (2011) 87-100.
[8] M. Dolecki, Tree Parity Machine synchronization time - statistical analysis, Mathematics, Physics and Informatics Series 6-153 (2012) 149-151.
[9] A. Gil, T. Karoń, Analysis of means and methods of protection the operating systems, Advances in Science and Technology 12 (2012) 149-168 (in Polish).
[10] M. Hassoun, Fundamentals of artificial neural networks, MIT Press, 1995.
[11] I. Kanter, W. Kinzel, The theory of neural networks and cryptography, Proceeding of the XXII Solvay Conference on Physics, The Physics of Communication, 2003, 631-644.
[12] I. Kanter, W. Kinzel, E. Kanter, Secure exchange of information by synchronization of neural networks, Europhys Letters 57 (2002) 141-147.
[13] I. Kanter, W. Kinzel, Neural cryptography, Proceedings of the 9th International Conference on Neural Information Processing 3 (2002) 1351-1354.
[14] E. Klein, R. Mislovaty, I. Kanter, A. Ruttor, W. Kinzel, Synchronization of neural networks by mutual learning and its application to cryptography, Advances in Neural Information Processing Systems, MIT Press Cambridge 17 (2005) 689-696.
[15] A. Klimov, A. Mityagin, A. Shamir, Analysis of neural cryptography, Proceedings of the 8th International Conference on the Theory and Application of Cryptology and Information Security ASIACRYPT'2002, Queenstown, 2003, 287-298.
[16] J. Konieczny, Application of the artificial neural networks for prediction of hardness of alloyed copper, Journal of Achievements in Materials and Manufacturing Engineering 55/2 (2012) 529-535.
[17] W. Kwaśny, W. Sitek, L.A. Dobrzański, Modelling of properties of the PVD coatings using neural networks, Journal of Achievements in Materials and Manufacturing, Engineering 24/2 (2007) 163-166.
[18] K. Lenik, Coat wear-resistant materials on the basis of Fe-Mn-C-B, National Academy of Ukrainian Science, Institute of Material Sciences, Lviv, 2004, 1-285.
[19] A. Menezes, S. Vanstone, P. Van Oorschot, Handbook of applied cryptography, CRC Press, 1996.
[20] S. Osowski, Neural networks in algorithmic approach, Publishing House WNT, 1996 (in Polish).
[21] A. Popko, K. Lenik, Integrated information systems, Information Technology in Teaching, Lublin Scientific Society LTN (2006) 127-133 (in Polish).
[22] L. Rutkowski, Methods and Technics of Artificial Intelligence, Publishing House PWN, Warsaw, 2006 (in Polish).
[23] A. Ruttor, Neural synchronization and cryptography, Ph.D. thesis, Wurzburg, 2006.
[24] A. Ruttor, W. Kinzel, R. Naeh, I. Kanter, Genetic attack on neural cryptography, Physical Review E 73-3 (2006) 036121-1-8.
[25] R. Rząsiński, Databases and computer programs selection of technological features, Journal of Achievements in Materials and Manufacturing Engineering 49/2 (2011) 350-359.
[26] J. Stokłosa, T. Bilski, T. Pankowski, Data Security in Informatical Systems, Publishing House PWN, 2001 (in Polish).