The Sound User Interface of Honeypot

Andrey Vishnevsky
Information Security Department
Bauman Moscow State Technical University
Moscow, Russia
anrdey.s.vishnevsky@gmail.com

Petr Klyucharev
Information Security Department
Bauman Moscow State Technical University
Moscow, Russia
pk.iu8@yandex.ru

***Abstract. Honeypots* are the traps disguised as information resources that record information about computer attacks. The collected data replenish the signature database of antiviruses and blacklists of firewalls, and serve as reliable evidence in computer forensics. All trapped files, Internet links, network addresses, and other artifacts are obviously fragments of malware since ordinary users do not interact with honeypots. This article developed the interface for working with a honeypot by ear. Attributes of computer attacks captured by the honeypot are encoded with sounds. The input alphabet for coding in the sound interface is the Cartesian product of analysis complexity, verdict, classification accuracy, and position of the compromise indicator in the list of attributes. The output alphabet is the set of piano key sounds. It is proposed to voice data letter by letter, with the possibility to re-record. The article proposes to use the audio interface in applications for the analysis of computer attacks, which can reduce the visual load for people with healthy eyes and allow professional work for the visually impaired. The proposed method for coding computer attack data allows performing part of a data security specialist's tasks by ear.**

Keywords – malicious code, reverse engineering programs, computer attack analysis, physical encoding, data listening, computer blind aid technologies, applications for disabled people.

I. INTRODUCTION

Honeypots are the traps for computer attacks that collect malicious objects for further analysis [1-9]. A large data stream forces the user of honeypots to work with multiple monitors at the same time or often switch between graphical interfaces on a single monitor.

As an alternative, a sound interface is proposed in this work that allows the user to navigate through a structured set of features and evaluate by ear the verdict of the current object, threat type, and calculation accuracy. The sound interface is based on the original method of computer attack feature encoding using the combinations of sounds at different frequencies.

II. SUBJECT AREA STATE

Information security experts find strings, checksums, signatures, names of operating system objects, and other features by which they determine the verdict of an object under investigation, i.e. the presence or absence of malicious functionality in a file, network request, email, website, etc. [10-15]. The accuracy of the attack classification indicates whether sufficient or only necessary features of a specific type of threat are found [24].

Scientific literature has publications on the ideas on the use of surround sound in applications for the visually impaired, and publications that mentioned operating systems with a limited set of application software that can be used by ear. However, there are no applications with an audio interface for information security experts [17-23]. This paper's aim is filling in the shortage of such tools.

III. PROBLEM FORMULATION

A feature of a computer attack is a tuple $\langle u,v,y,z\rangle$ containing the name of a potentially dangerous action $u \in U$, Boolean verdict $v \in Z_2$, tool name $y \in Y$ with which the feature was obtained, and Boolean value of the classification accuracy $z \in Z_2$, which implies the verdict confidence or uncertainty. U and Y are the lists of potentially dangerous actions and analysis tools listed in the MITRE ATT & CK computer attack encyclopaedia.

It is required to present a comprehensive set of computer attacks in order to provide navigation by ear and auditory recognition of the verdict, threat type, attack attributes, and calculated classification accuracy.

IV. SOLUTION

We will group all analysis tools into reputational ones–those not requiring the presence of an investigated file, static ones–those not requiring a file to run, dynamic ones–those requiring a file to run in a controlled environment, and hybrid ones that include all the above types of analysis [12]. We divide the set of collected computer attack features into four lists, depending on the complexity of the analysis to which the y tool belongs.

$$U = k \in L_4 U_k, L_4 = \{0,1,2,3\}, \forall y \in U \qquad (1)$$

We divide each list into twelve parts and assign the i^{th} attribute the index j equal to the number of the part in which it is contained:

$$j = \left\lfloor \frac{12i}{n} \right\rfloor, j \in \{0,1,2,\ldots,11\}, \qquad (2)$$

where n is the length of the feature list.

The user can navigate through the feature list by using the keyboard. By clicking the up-arrow (down-arrow), the user can increase (decrease) the number of the current feature *i*. Scrolling is cyclical so from the last part one gets to the first one. Index *j* is incremented when the current feature reaches such value i' that

$$i': \left\lfloor \frac{12i'}{n} \right\rfloor > \left\lfloor \frac{12(i'-1)}{n} \right\rfloor \tag{3}$$

Navigation through the twelfth parts of features can be presented as a cyclical group $(Z_{12}, +)$ with respect to the operation of moving to the twelfth part of the feature list which is some PageDown (PageUp) away from the current part. No more than 11 clicks per unit time are taken into account. PageDown (PageUp) clicks increase (decrease) the number of the displayed part and move *I* to the beginning of another twelfth part of the list j' depending on the number of clicks *k*:

$$\begin{cases} j': (j+k) \ mod \ 12 & (4) \\ i': \left\lceil \frac{n}{12} \cdot j' \right\rceil, & (5) \end{cases}$$

where $j, j', k \in Z_{12}$.

When you scroll through the list of features, the number of the current feature *i* is incremented. Scrolling through the list is cyclical, i.e. from the last part, we go to the first one. The index *j* is incremented when the number of the current attribute reaches the value *i'*, such that

$$i': \left\lfloor \frac{12i'}{n} \right\rfloor > \left\lfloor \frac{12(i'-1)}{n} \right\rfloor \tag{6}$$

Let's represent the set of indices *j* in the form of a cyclic group $(Z_{12}, +)$ with respect to the operation of transition to the next twelfth of the list of attributes.

The input alphabet *A* for encoding in the sound interface is the Cartesian product of analysis complexity L_4, position in the list N_{12}, verdict K_2, and classification accuracy M_2 [25].

$$A = L_4 \times N_{12} \times K_2 \times M_2 \tag{7}$$

Table I shows an example of a list of 30 features. Six parts contain three elements, and the other six parts contain two elements 3*6 + 2*6 = 30. The index *j* is the number to which the ordinal number belongs.

The output alphabet *B* is the set of piano key sounds. The sound wave generated by pressing the piano key is described by the sum of harmonic oscillations:

$$x = \sum_{k=1}^{n} x_{0k} sin(\omega_k t + \varphi_{0k}), \tag{8}$$

where x_{0k} is the amplitude, ω_k is the circular frequency, φ_{0k} is the initial phase of the k^{th} component.

TABLE I. INDEXED LIST OF COMPUTER ATTACK ATTRIBUTES

i	**Malicious Action**	**Index *j***
0	Data Encrypted for Impact: Chrysanthemum.jpg.crypto	0
1	Data Encrypted for Impact: Desert.jpg.crypto	0
2	Data Encrypted for Impact: Hydrangeas.jpg.crypto	0
3	Data Encrypted for Impact: Jellyfish.jpg.crypto	1
4	Data Encrypted for Impact: Koala.jpg.crypto	1
5	Data Encrypted for Impact: Lighthouse.jpg.crypto	2
6	Data Encrypted for Impact: Penguins.jpg.crypto	2
7	Data Encrypted for Impact: Tulips.jpg.crypto	2
8	Data Encrypted for Impact: DSC_1001.JPG.crypto	3
9	Data Encrypted for Impact: DSC_1002.JPG.crypto	3
10	Data Encrypted for Impact: DSC_1003.JPG.crypto	4
11	Data Encrypted for Impact: DSC_1004.JPG.crypto	4
12	Data Encrypted for Impact: DSC_1005.JPG.crypto	4
13	Registry Run Keys: CurrentVersion/Run	5
14	Data Encrypted for Impact: Passport.jpg.crypto	5
15	Data Encrypted for Impact: Scan.jpg.crypto	6
16	Data Encrypted for Impact: Payment.jpg.crypto	6
17	Registry Run Keys: Startup Folder	6
18	Data Encrypted for Impact: Roadmap.jpg.crypto	7
19	Data Encrypted for Impact: Task.jpg.crypto	7
20	Data Encrypted for Impact: Workflow.jpg.crypto	8
21	Data Encrypted for Impact: Pipeline.jpg.crypto	8
22	Data Encrypted for Impact: Collaborators.jpg.crypto	8
23	Registry Run Keys: CurrentVersion/RunOnce	9
24	Registry Run Keys: CurrentVersion/RunOnceEx	9
25	Data Encrypted for Impact: December.jpg.crypto	10
26	PowerShell	10
27	Injection of Dynamic-link library: Evil.dll	10
28	Registry Run Keys: Winlogon/Notify	11
29	Registry Run Keys: Explorer\Browser Helper Objects	11

All musical sounds of the piano keys are divided into octaves and have the following letter designations: “c” for Do, “d” for Re, “e” for Mi, “f” for Fa, “g” for Sol, “a” for La, “b” for A sharp, “h” for C [25, 29]. In a contraoctave, notes are denoted by capital letters with a subscript 1, in a small octave by capital letters without an index, and in the octaves 1-5 by lowercase letters with a superscript corresponding to the octave number. Notes a half tone higher have the suffix "*is*", and for half-tone-lowered notes, the suffix "*es*" is used. The subjective perception of a pitch is determined by the root frequency ω_0 ω_0 [24]. The piano keyboard contains eight full octaves, at the borders of which the pitch frequency ω_0 is doubled (see Tables II and III). Each octave is represented by twelve keys with a constant ratio of adjacent key sounds equal to $\omega_0^i/\omega_0^{i+1} = 2^{1/12}$ $\omega_0^i/\omega_0^{i+1} = 2^{1/12}$.

TABLE II. TONE FREQUENCIES IN LOWER OCTAVES

Tone	Subcontra octave		Contra octave		Big Octave		Small Octave	
	Designation	Frequency, Hz	Designation	Frequency, Hz	Designation	Frequency, Hz	Designation	Frequency, Hz
C	C_2	16,3	C_1	32,7	C	65,4	c	130
C sharp (D flat)	Cis_2	17,3	Cis_1	34,6	Cis	69,3	cis	138
D	D_2	18,3	D_1	36,7	D	73,4	d	146
D sharp (E Flat)	Dis_2	19,4	Dis_1	38,8	Dis	77,7	dis	155
E	E_2	20,6	E_1	41,2	E	82,4	e	164
F	F_2	21,8	F_1	43,6	F	87,3	f	174
F sharp (G flat)	Fis_2	23,1	Fis_1	46,2	Fis	92,5	fis	185
G	G_2	24,5	G_1	49	G	98	g	196
G sharp (A flat)	Gis_2	24,5	Gis_1	51,9	Gis	103	gis	207
A	A_2	27,5	A_1	55	A	110	a	220
A sharp (B flat)	B_2	29,1	B_1	58,2	B	116	b	233
B	H_2	30,8	H_1	61,7	H	123	h	246

Def. The minor scale is a sequence of seven sounds of the same octave, such that the ratio of the frequencies of the second and third sounds (as well as that of the fifth and sixth) makes $2^{1/12}$, while the ratio of the frequencies of other neighboring sounds makes $2^{1/6}$ [32, 37, 39].

Def. The major scale is a sequence of seven sounds of the same octave, such that the ratio of the frequencies of the third and fourth sounds (as well as that of the seventh and first of the next octave) is $2^{1/12}$, and the ratio of the frequencies of other neighboring sounds makes $2^{1/6}$ [32, 37, 39].

Def. Transposition by *n* degrees means the increase of the frequency of each sound in a sequence of sounds by $2^{n/12}$ times [32-34, 38].

$$T_n(x) = x \cdot 2^{n/12} \tag{9}$$

$$T_n(\langle x_1, x_2, \ldots, x_m \rangle) = \langle T_n(x_1), T_n(x_2), \ldots, T_n(x_m) \rangle, \tag{10}$$

where $\langle x_1, x_2, \ldots, x_m \rangle$ is the sequence of sounds.

It is possible to determine by ear which octave and scale a sequence of sounds belongs to, how many degrees it was transposed by, and whether it is ended by the same sound from which it began [27, 45]. Since doubling the frequency of sound during transposition by an integer number of octaves is subjectively perceived as the same sound, the scale is described by a cyclic group (Z_{12}, T_1) regarding the operation of transposition by one degree [28-39, 43, 44].

TABLE III. SCALE OF TONE FREQUENCIES IN THE UPPER OCTAVES

Tone	First octave		Second octave		Third octave		Fourth octave	
	Designation	Frequency, Hz	Designation	Frequency, Hz	Designation	Frequency, Hz	Designation	Frequency, Hz
C	c^1	261	c^2	523	c^3	1046	c^4	2093
C sharp (D flat)	cis^1	277	cis^2	554	cis^3	1108	cis^4	2217
D	d^1	293	d^2	587	d^3	1174	d^4	2349
D sharp (E flat)	dis^1	311	dis^2	622	dis^3	1244	dis^4	2489
E	e^1	329	e^2	659	e^3	1318	e^4	2637
F	f^1	349	f^2	698	f^3	1396	f^4	2793
F sharp (G flat)	fis^1	369	fis^2	739	fis^3	1479	fis^4	2959
G	g^1	392	g^2	783	g^3	1567	g^4	3135
G sharp (A flat)	gis^1	415	gis^2	830	gis^3	1661	gis^4	3322
A	a^1	440	a^2	880	a^3	1760	a^4	3520
A sharp (B flat)	b^1	466	b^2	932	b^3	1864	b^4	3729
B	h^1	493	h^2	987	h^3	1975	h^4	3951

Let us take the output alphabet *B* as the Cartesian product of the sets of octaves O_8, initial degrees S_{12}, scales L_2, and the set of sequences completed and not completed with the original sound P_2.

$$B = O_8 \times S_{12} \times L_2 \times P_2 \tag{11}$$

Let's develop the isomorphism $\varphi: A \to B$. We associate a minor sound sequence to each feature of the malicious code *u*, and a major one to a harmless code feature. The complexity of the analysis is comparable to an octave, accuracy is comparable to the completeness of the sequence of sounds, and the position in the list is comparable to the first degree of the sound sequence.

$$\varphi(\langle u, v, y, z, pos \rangle) = T_{-12 \cdot y} \circ T_{pos}\left(sound_{u,v}(z)\right) \tag{12}$$

Tables IV and V show an example of an encoded feature of a computer attack.

TABLE IV. ATTACK FEATURE, ENCODED DEPENDING ON LOCATION AND ANALYSIS TYPE

Position	**Analysis Level**	
	Hybrid	***Dynamic***
0	$\langle d, g, fis, g, d\rangle$	$\langle d^1, g^1, fis^1, g^1, d^1\rangle$
1	$\langle d, g, fis, g, d\rangle$	$\langle d^1, g^1, fis^1, g^1, d^1\rangle$
2	$\langle e, a, gis, a, e\rangle$	$\langle e^1, a^1, gis^1, a^1, e^1\rangle$
3	$\langle f, b, a, b, f\rangle$	$\langle f^1, b^1, a^1, b^1, f^1\rangle$
4	$\langle fis, h, b, h, fis\rangle$	$\langle fis^1, h^1, b^1, h^1, fis^1\rangle$
5	$\langle g, c^1, h, c^1, g\rangle$	$\langle g^1, c^2, h^1, c^2, g^1\rangle$
6	$\langle gis, cis^1, c^1, cis^1, gis\rangle$	$\langle gis^1, cis^2, c^2, cis^2, gis^1\rangle$
7	$\langle gis, cis^1, c^1, cis^1, gis\rangle$	$\langle gis^1, cis^2, c^2, cis^2, gis^1\rangle$
8	$\langle b, dis^1, d^1, dis^1, b\rangle$	$\langle b^1, dis^2, d^2, dis^2, b^1\rangle$
9	$\langle h, e^1, dis^1, e^1, h\rangle$	$\langle h^1, e^2, dis^2, e^2, h^1\rangle$
10	$\langle c^1, f^1, e^1, f^1, c^1\rangle$	$\langle c^2, f^2, e^2, f^2, c^2\rangle$
11	$\langle cis^1, fis^1, f^1, fis^1, cis^1\rangle$	$\langle cis^2, fis^2, f^2, fis^2, cis^2\rangle$

TABLE V. ATTACK FEATURE, ENCODED DEPENDING ON LOCATION AND ANALYSIS TYPE

Position	**Analysis Level**	
	Static	***Reputational***
0	$\langle d^2, g^2, fis^2, g^2, d^2\rangle$	$\langle d^3, g^3, fis^3, g^3, d^3\rangle$
1	$\langle d^2, g^2, fis^2, g^2, d^2\rangle$	$\langle d^3, g^3, fis^3, g^3, d^3\rangle$
2	$\langle e^2, a^2, gis^2, a^2, e^2\rangle$	$\langle e^3, a^3, gis^3, a^3, e^3\rangle$
3	$\langle f^2, b^2, a^2, b^2, f^2\rangle$	$\langle f^3, b^3, a^3, b^3, f^3\rangle$
4	$\langle fis^2, h^2, b^2, h^2, fis^2\rangle$	$\langle fis^3, h^3, b^3, h^3, fis^3\rangle$
5	$\langle g^2, c^3, h^2, c^3, g^2\rangle$	$\langle g^3, c^4, h^3, c^4, g^3\rangle$
6	$\langle gis^2, cis^3, c^3, cis^3, gis^2\rangle$	$\langle gis^3, cis^4, c^4, cis^4, gis^3\rangle$
7	$\langle gis^2, cis^3, c^3, cis^3, gis^2\rangle$	$\langle gis^3, cis^4, c^4, cis^4, gis^3\rangle$
8	$\langle b^2, dis^3, d^3, dis^3, b^2\rangle$	$\langle b^3, dis^4, d^4, dis^4, b^3\rangle$
9	$\langle h^2, e^3, dis^3, e^3, h^2\rangle$	$\langle h^3, e^4, dis^4, e^4, h^3\rangle$
10	$\langle c^3, f^3, e^3, f^3, c^3\rangle$	$\langle c^4, f^4, e^4, f^4, c^4\rangle$
11	$\langle cis^3, fis^3, f^3, fis^3, cis^3\rangle$	$\langle cis^4, fis^4, f^4, fis^4, cis^4\rangle$

At any time, a user listens to one sound recording. Then they move on to the next sound recording, i.e. the next feature. It is important that by increasing the pitch one can understand how many elements are left in the list. An exceptional case will be with only one attribute in the list. To distinguish a list with one attribute from a list with an excessively large number of attributes, one can introduce an additional sound signal.

After listening to the sound sequence, one can request a verbal explanation. If an explanation is not contained in the dictionary, the interface will spell it out. The user can voice any spelled or dictionary explanation independently by recording their speech using a microphone.

V. CONCLUSIONS

The proposed method for sound encoding of computer attack data allows one to perform part of a virus analyst's tasks by ear. This reduces eye fatigue and opens up the possibility of professional work for the visually impaired. In the future, it will be possible to refine the encoding method using surround sound to display a large number of objects.

REFERENCES

[1] Vishnevsky A.S. Content based attack detection in web-oriented honeypots. Voprosy kiberbezopasnosti [Cybersecurity issues], 2018. No 3. P. 54-62. DOI: https://doi.org/10.21681/2311-3456-2018-3-54-62

[2] Bilinski M., Gabrys R., Mauger J. Optimal Placement of Honeypots for Network Defense. In: Bushnell L., Poovendran R., Başar T. (eds) Decision and Game Theory for Security. GameSec 2018. Lecture Notes in Computer Science, vol 11199. Springer, Cham. 2018. DOI: https://doi.org/10.1007/978-3-030-01554-1_7

[3] Chee Keong NG, Lei Pan, Dr. Yang Xiang. Honeypot Frameworks and Their Applications: A New Framework. In Springer Briefs on Cyber Security Systems and Networks. Springer, Singapore. 2018. DOI: https://doi.org/10.1007/978-981-10-7739-5

[4] Papazis, K. & Chilamkurti. Detecting indicators of deception in emulated monitoring systems. Service Oriented Computing and Applications. 2019. Vol. 13. P. 17-29. DOI: https://doi.org/10.1007/s11761-018-0252-2

[5] Sadasivam G.K., Hota C., Anand B. Honeynet Data Analysis and Distributed SSH Brute-Force Attacks. In: Chakraverty S., Goel A., Misra S. (eds) Towards Extensible and Adaptable Methods in Computing. Springer, Singapore. 2018. DOI: https://doi.org/10.1007/978-981-13-2348-5_9

[6] Voronchikhin I., Ivanov I., Maximov R., Sokolovsky S. Masking of Distributed Information Systems Structure in Cyberspace. Voprosy kiberbezopasnosti [Cybersecurity issues], 2019. No 6 (34), pp. 92-101. DOI: 10.21681/2311-3456-2019-6-92-101.

[7] Wang, Y., Guo, Y., Zhang, L. et al. SWIM: An Effective Method to Perceive Cyberspace Situation from Honeynet. Arabian Journal for Science and Engineering. 2018. Vol. 43. P. 6863. DOI: https://doi.org/10.1007/s13369-017-2904-5

[8] Yazov Yu., Serdechnyy A., Sharov I. Methodical approach for estimation of efficiency of honeypot system. Voprosy kiberbezopasnosti [Cybersecurity issues], 2014, No 1 (2), pp. 55-60.

[9] Yin W., Zhou H., Yang C. (2018) A Honeyfarm Data Control Mechanism and Forensic Study. In: Li B., Shu L., Zeng D. (eds) Communications and Networking. ChinaCom 2017. Lecture Notes of the Institute for Computer Sciences, Social Informatics and Telecommunications Engineering, vol 237. Springer, Cham. DOI: https://doi.org/10.1007/978-3-319-78139-6_37

[10] Keim, Y., Mohapatra, A.K. Cyber threat intelligence framework using advanced malware forensics. International Journal of Information Technology. Springer Singapore. 2019. DOI: https://doi.org/10.1007/s41870-019-00280-3

[11] Zhang, H., Yi, Y., Wang, J. et al. Network attack prediction method based on threat intelligence for IoT. Multimedia Tools and Applications. Springer US. 2019. Vol 78. P. 30257–30270. DOI: https://doi.org/10.1007/s11042-018-7005-2

[12] Lu X., Han J., Ren Q., Dai H., Li J., Ou, J. Network threat detection based on correlation analysis of multi-platform multi-source alert data. Multimedia Tools and Applications. 2018. DOI: https://doi.org/10.1007/s11042-018-6689-7

[13] Böhm F., Menges F., Pernul G. Graph-based visual analytics for cyber threat intelligence. Cybersecurity. 2018. Vol 1. P. 16. DOI: https://doi.org/10.1186/s42400-018-0017-4u

[14] Barabanov A., Grishin M., Markov A., Tsirlov V. Current Taxonomy of Information Security Threats in Software Development Life Cycle. In: 2018 IEEE 12th International Conference Application of Information and Communication Technologies (AICT). IEEE (17-19 Oct 2018, Almaty, Kazakhstan). 2018, pp. 356-361. DOI: 10.1109/ICAICT.2018.8747065.

[15] Yu B., Fang Y., Yang Q., Tang Y., Liu, L. (2018). A survey of malware behavior description and analysis. Frontiers of Information Technology & Electronic Engineering, 19(5), 583–603. DOI: https://doi.org/10.1631/FITEE.1601745

[16] Monnappa K. A. Learning Malware Analysis. Explore the concepts, tools, and techniques to analyze and investigate Windows malware. Packt Publishing, 2018. ISBN 978-1-78839-250-1

[17] Frauenberger C., Noisternig M. 3D Audio Interfaces for the Blind. Proceedings of the 2003 International Conference on Auditory Display, Boston, MA, USA, July 6-9, 2003

[18] Belokon' M.V. Komp'juternye tiflosredstva kak faktor obespechenija bezbar'ernoj sredy dlja slepyh i slabovidjashhih v vysshem inkljuzivnom obrazovanii. Komp'juternye nauki i informacionnye tehnologii Materialy Mezhdunarodnoj nauchnoj konferencii. Izdat. centr Nauka. Saratov. 2016. P. 63-67. ISBN 978-5-9999-2651-7

[19] Trunova L. V., Grechuhina I. A. Organizacija programmno-apparatnoj sistemy po soprovozhdeniju obrazovatel'noj dejatel'nosti v uchrezhdenii professional'nogo obrazovanija dlja obuchajushhihsja s ogranichennymi vozmozhnostjami zdorov'ja // Molodoj uchenyj. — 2018. — №24. — P. 324-335. ISSN: 2072-0297

[20] Khan, A., Khusro, S., Niazi, B. et al. TetraMail: a usable email client for blind people. Universal Access in the Information Society. 2018. Pp. 1-20. DOI: https://doi.org/10.1007/s10209-018-0633-5

[21] Khan, A., Khusro, S. Blind-friendly user interfaces – a pilot study on improving the accessibility of touchscreen interfaces. Multimedia Tools and Applications. 2019. Vol. 78. Pp. 17495-17519. DOI: https://doi.org/10.1007/s11042-018-7094-y

[22] Červenka P., Břinda K., Hanousková M., Hofman P., Seifert R. (2016) Blind Friendly Maps. In: Miesenberger K., Bühler C., Penaz P. (eds) Computers Helping People with Special Needs. ICCHP 2016. Lecture Notes in Computer Science, vol 9759. Springer, Cham. DOI: https://doi.org/10.1007/978-3-319-41267-2_18

[23] Bhowmick, A., Hazarika, S.M. An insight into assistive technology for the visually impaired and blind people: state-of-the-art and future trends. Journal on Multimodal User Interfaces. 2017. Vol 11. Pp. 149-172. DOI: https://doi.org/10.1007/s12193-016-0235-6

[24] Tom M. Mitchell. Machine Learning. McGraw-Hill Science/Engineering/Math. 1977. ISBN: 0070428077

[25] Kientzle T. A Programmer's Guide to Sound. Addison-Wesley. 1997. ISBN 5-94074-014-6

[26] Sidel'nikov V.M. Teorija kodirovanija.: Uchebnoe posobie. – M.:Fizmatlit, 2008. – Pp. 324. ISBN: 978-5-9221-0943-7

[27] Kuznecov L.A. Akustika muzykal'nyh instrumentov: Sprav.— M.: Legprombytizdat, 1989. – Pp. 389. ISBN 5-7088-0166-2.

[28] Berry C., Fiore T. Hexatonic Systems and Dual Groups in Mathematical Music Theory. Involve, a Journal of Mathematics. 2016. DOI: https://doi.org/10.2140/involve.2018.11.253

[29] Hughes J.R. Using Fundamental Groups and Groupoids of Chord Spaces to Model Voice Leading. In: Collins T., Meredith D., Volk A. (eds) Mathematics and Computation in Music. MCM 2015. Lecture Notes in Computer Science, vol 9110. Springer, Cham. DOI: https://doi.org/10.1007/978-3-319-20603-5_28

[30] Peck R.W. All-Interval Structures. In: Collins T., Meredith D., Volk A. (eds) Mathematics and Computation in Music. MCM 2015. Lecture Notes in Computer Science, vol 9110. Springer, Cham. DOI: https://doi.org/10.1007/978-3-319-20603-5_29

[31] Schlemmer T. Unifying Tone System Definitions: Ordering Chromas. In: Collins T., Meredith D., Volk A. (eds) Mathematics and Computation in Music. MCM 2015. Lecture Notes in Computer Science, vol 9110. Springer, Cham. DOI: https://doi.org/10.1007/978-3-319-20603-5_30

[32] Popoff A., Andreatta M., Ehresmann A. A Categorical Generalization of Klumpenhouwer Networks. In: Collins T., Meredith D., Volk A. (eds) Mathematics and Computation in Music. MCM 2015. Lecture Notes in Computer Science, vol 9110. Springer, Cham. DOI: https://doi.org/10.1007/978-3-319-20603-5_31

[33] Amiot E. Can a Musical Scale Have 14 Generators?. In: Collins T., Meredith D., Volk A. (eds) Mathematics and Computation in Music. MCM 2015. Lecture Notes in Computer Science, vol 9110. Springer, Cham. DOI: https://doi.org/10.1007/978-3-319-20603-5_35

[34] Booth M.D. Primal-Circular Substitutions. In: Agustín-Aquino O., Lluis-Puebla E., Montiel M. (eds) Mathematics and Computation in Music. MCM 2017. Lecture Notes in Computer Science, vol 10527. Springer, Cham. DOI: https://doi.org/10.1007/978-3-319-71827-9_1

[35] Cannas S., Antonini S., Pernazza L. On the Group of Transformations of Classical Types of Seventh Chords. In: Agustín-Aquino O., Lluis-Puebla E., Montiel M. (eds) Mathematics and Computation in Music. MCM 2017. Lecture Notes in Computer Science, vol 10527. Springer, Cham. DOI: https://doi.org/10.1007/978-3-319-71827-9_2

[36] Genuys G., Popoff A. Homometry in the Dihedral Groups: Lifting Sets from Zn to Dn. In: Agustín-Aquino O., Lluis-Puebla E., Montiel M. (eds) Mathematics and Computation in Music. MCM 2017. Lecture Notes in Computer Science, vol 10527. Springer, Cham. DOI: https://doi.org/10.1007/978-3-319-71827-9_4

[37] Peck R.W. Almost Difference Sets in Transformational Music Theory. In: Agustín-Aquino O., Lluis-Puebla E., Montiel M. (eds) Mathematics and Computation in Music. MCM 2017. Lecture Notes in Computer Science, vol 10527. Springer, Cham. DOI: https://doi.org/10.1007/978-3-319-71827-9_6

[38] Agustín-Aquino O.A., Mazzola G. Contrapuntal Aspects of the Mystic Chord and Scriabin's Piano Sonata No. 5. In: Montiel M., Gomez-Martin F., Agustín-Aquino O. (eds) Mathematics and Computation in Music. MCM 2019. Lecture Notes in Computer Science, vol 11502. Springer, Cham. DOI: https://doi.org/10.1007/978-3-030-21392-3_1

[39] Popoff A., Andreatta M., Ehresmann A. Groupoids and Wreath Products of Musical Transformations: A Categorical Approach from poly-Klumpenhouwer Networks. In: Montiel M., Gomez-Martin F., Agustín-Aquino O. (eds) Mathematics and Computation in Music. MCM 2019. Lecture Notes in Computer Science, vol 11502. Springer, Cham. DOI: https://doi.org/10.1007/978-3-030-21392-3_3

[40] Noll T., Clampitt D. Exploring the Syntonic Side of Major-Minor Tonality. In: Montiel M., Gomez-Martin F., Agustín-Aquino O. (eds) Mathematics and Computation in Music. MCM 2019. Lecture Notes in Computer Science, vol 11502. Springer, Cham. DOI: https://doi.org/10.1007/978-3-030-21392-3_10

[41] Jedrzejewski F. Non-Contextual JQZ Transformations. In: Montiel M., Gomez-Martin F., Agustín-Aquino O. (eds) Mathematics and Computation in Music. MCM 2019. Lecture Notes in Computer Science, vol 11502. Springer, Cham. DOI: https://doi.org/10.1007/978-3-030-21392-3_12

[42] del Pozo I., Gómez F. Formalization of Voice-Leadings and the Nabla Algorithm. In: Montiel M., Gomez-Martin F., Agustín-Aquino O. (eds) Mathematics and Computation in Music. MCM 2019. Lecture Notes in Computer Science, vol 11502. Springer, Cham. DOI: https://doi.org/10.1007/978-3-030-21392-3_30

[43] Ebeling, M.: Neuronal periodicity detection as a basis for the perception of consonance: a mathematical model of tonal fusion. J. Acoust. Soc. Am. 124, 2320 (2008). DOI: https://doi.org/10.1121/1.2968688

[44] Bowling D., Purves D. A biological rationale for musical consonance. Proc Natl Acad Sci USA. 2015. 112(36):11155–11160. DOI: https://doi.org/10.1073/pnas.1505768112

[45] Ravignani A., Bowling D., Fitch W. Chorusing, synchrony, and the evolutionary functions of rhythm. Frontiers in Psychology. Vol. 5. DOI: https://doi.org/10.3389/fpsyg.2014.01118