Designing Mazes for 2D Games by Artificial Ant Colony Algorithm

Dawid Połap
Institute of Mathematics
Silesian University of Technology
Kaszubska 23, 44-100 Gliwice, Poland
Email: Dawid.Polap@gmail.com

***Abstract*—In this paper a novel approach to designing boards for 2D games is proposed. The author proposes a solution based on application of Computational Intelligence to create mazes used for entertainment like "find the exit games" or as an environment for other popular 2D games. The use of Computational Intelligence in the form of adapting the behavior of ants to build roads in the map is based on the phenomenon of leaving pheromones while searching for the source of the food. In order to adapt the algorithm to create mazes, author presents the march of the queen as a condition for stopping the algorithm. Experiments have been performed on various shape and dimensions of mazes, to prove effectiveness and speed of the presented method and its many advantages.**

I. Introduction

Nowadays, CI is in its heyday. Computational Intelligence (CI) methods are used in almost every field of Information Science. Behind this success is an innovation, speed of action or simplicity of implementing and multifarious adaptability. For instance, in [16], [22], [23] presented the application of various methods of Evolutionary Computation for finding key-points on 2D images for classification and recognition purpose [1], [2]. In [30], the authors proposed a novel approach to automatic medical signal diagnosis. Their research showed that CI solutions can be used to retrieve missing or incomplete medical data and used as a training set in neural network to recognize health threats. In [39] presented the use of CI methods to create a classifier analysing employees to form work groups using a probabilistic neural network. Again in [26], [31] described the use of swarm intelligence algorithms in the positioning and queuing systems through the use of dedicated fitness function. In [12] presented a new adaptive technique for image compression which uses a Discrete Wavelet Transform and Radial Basis Function Neural Networks [3], [4], [5], [6]. Again, in [19], CI methods is used for graphics processing such as image edge detection, in [42] showed using CI and histograms for image segmentation and in [24] described the application of firefly algorithm and cuckoo search algorithm in multilevel image thresholding. The authors [17] showed application of an evolutionary algorithm for finding the optimal scaling factors in digital image watermarking. CI algorithms find their utility also in image compression algorithms (see [41], where the authors compare different heuristic algorithms used to vector quantization). In games CI has various applications, i.e. [36] proposed general game playing player which is based on selecting the most attractive strategy for spesific game, in [15] described an attempt at the implementation an agent based on tree search methods. Interesting topic in which CI found considerable application is assessment the playability of games eg.: [27], [40].

Logic puzzles are an integral part of entertainment which find place in various newspapers, magazines and even mobile applications or web pages. These types of tasks are to find a solution or answer by using reasoning based on knowledge or intuition. In [35], the authors describe two directions of game design courses. Depending on the specific objectives of the puzzle, the solution may require time and patience. The main purpose of such puzzles is not only entertainment but also teaching by increase our knowledge and stimulate curiosity. For example, in the case of crossword puzzle, the purpose is to find the hidden password. Another example is the maze. Maze is a system constructed of many different paths of which only a few lead to the exit. The problem is to find the way through the maze from entrance to the exit. Besides the obvious use of mazes as riddle, it has also many applications in games of all kind. For instance, in 2D games like Pacman, where each level is another board. The board in such a game is nothing but a maze, where some rules were assumed during the construction of the board. The authors [10] introduced a new concept in robot-maze solving, where they presented the use of Modified Following Wall Navigation algorithm to navigate its way in virtual mazes. Similarly, the crossword puzzles are some kind of mazes. The creation of the generator allow us to create an infinite number of different combinations of mazes. As a result, it could become an additional mine of different environment for many games. For instance, in [29] several algorithms for designing mazes based on images were presented. The increased interest in the game Pacman among researchers and computer users brought back the era of Arcade Games that after many years return, but in a computerized form, resulting in greater demand for new game levels (most of these games uses simple boards like mazes) and artificial intelligence engines to improve the quality of the game. For example in [9], [21], [34], the authors describes a learning version of Pacman game using different methods ie.: artificial neural networks supporting a mathematical models (as in [7] or [8]). The authors [11] used a genetic algorithm for procedural generation of levels for platform games, and in [37] introduced an evaluation platform for general agents called the Arcade Learning Environmnet.

A. A brief history of mazes and their applications

First mazes appeared in ancient times the mythical labyrinth of King Minos is one of the most well-known. According to Cretan myths, labyrinth was designed by Daedalus to hide the Minotaur, which was a child of Kings wife and a bull. The monster was killed by Theseus who entered the maze and escaped through the abandoned thread. To the present day, there are no records about the shape of the maze but many researches identify a maze with caves in Gortyna in Crete. Another well-known labyrinth is the Great Labyrinth in Egypt which was built for 365 years. According to the records of the Greek historian Herodotus, labyrinth was a structure built on two levels, stretching for over 25 kilometers with many moving walls. Currently, archaeologists are looking for the remains of mazes, but it is difficult because it is believed that they are hidden under the 20-meter layer of sand (see [38]).

Besides buildings, mazes are very often used as a motif in art. For example, the Romans create mosaic labyrinths and vase painting. In the Gothic era, labyrinths appeared in churches and cathedrals, mainly in the form of floor of the nave. It was believed that the passage through the maze replaces the long pilgrimage.

Furthermore, hedgerow structures are built as mazes from Renaissance to the present day. The purpose of creating such mazes was primarily aesthetics and beauty of the gardens. Another advantage was the ability to search hidden items such as fountains placed in the middle of maze. Nowadays, labyrinths are used only in various games and aesthetic motifs.

B. Basics of designing mazes

Designing the maze requires a few basic rules that should be followed during its creation. At the beginning, it is necessary to choose the size of the maze, shape (i.e.: square, rectangular), the number of entrances/exists and their locations. Then, create a maze. An important aspect is the difficulty of navigating a maze that will define blind alleys and winding roads.

One of the most known approaches to create mazes is to apply graph theory. In [33], the author presents the greedy algorithm, which seeks tree containing vertices of the smallest weight. Kruskals algorithm is used to create labyrinths using the set of points and weights. In [14], the authors showed algorithm called Hunt-and-Kill maze generating algorithm, which works by carvings the walls. The algorithm randomly moves from one cell to the neighboring one and removes the wall on the basis of assumptions. Another methods is shown in [28] which creates mazes based on a rectangular black-and-white raster image. For comparison, in [13] presented an algorithm based on steganographic method which is an improved version of the algorithm presented in [14]. The described improvement is to consider multipaths to gain embedding capacity.

In this paper, I would like to present an alternative method for creating mazes based on Computational Intelligence. CI is considered as a modified version of the heuristic algorithm - Artificial Ant Colony Algorithm.

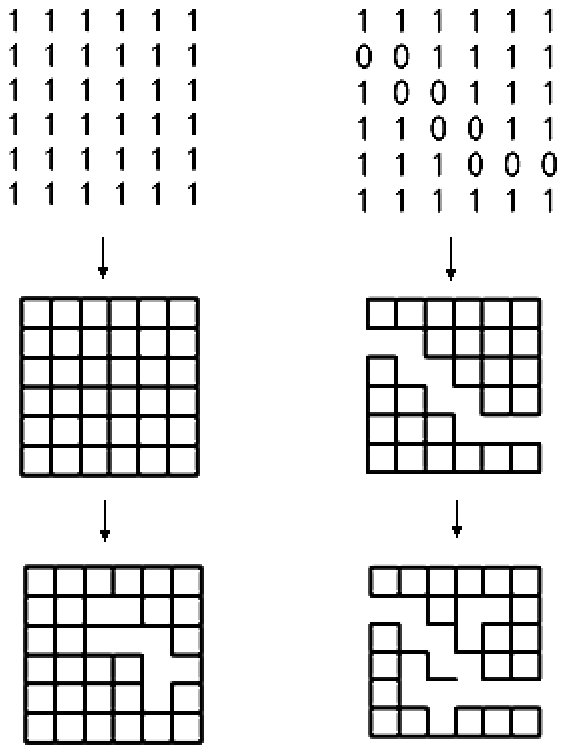


Fig. 1: The construction mazes from two different arrays of pheromones for $r = 6$.

II. Artificial Ant Colony Algorithm

A. Artificial Ant Colony Algorithm

Artificial Ant Colony Algorithm (AACA) maps behavior of ants while searching for the source of food. One of the first version of AACO was presented in [20] and [25] for optimization purpose.

Ants move randomly leaving a pheromone. Left pheromones create a trail of the pheromones that allows an access to the sought source of food. If the food is found by an ant, the ant returns home leaving the larger trail of the pheromone. As a result, another ants will be able to reach the source because the ant decides on its movement by choosing the place where the level of pheromones is significant.

At the beginning, the level of pheromone is everywhere the same. Then, after starting a new iteration it is updated in accordance with

$$f^{t+1}(\mathbf{x_i}, \mathbf{x_j}) = (1-\rho) f^t(\mathbf{x_i}, \mathbf{x_j}) + \Gamma_i^t, \tag{1}$$

where ρ means evaporation rate, t is the number of iterations and n is the number of insects in population that must come to an ant $\mathbf{x_i}$ over Γ_i^t distance, which is defined as

$$\Gamma_i^t = \sum_{i=1}^{n} \frac{1}{L_{ij}^t}, \tag{2}$$

where L_{ij}^t is the length of the road from ant i to ant j. The path length L_{ij} between the two ants i and j located at the points

$\mathbf{x}_i$ and $\mathbf{x}_j$ on the coordinate system is defined as Cartesian metric

$$L_{ij} = \|\mathbf{x}_i - \mathbf{x}_j\| = \sqrt{\sum_{k=1}^{2}(x_{i,k} - x_{j,k})^2}, \tag{3}$$

where $\mathbf{x}_i$ and $\mathbf{x}_j$ are points in $R \times R$ space, $x_{i,k}, x_{k,j}$-k-th components of the spatial coordinates $\mathbf{x}_i$ and $\mathbf{x}_j$ representing ants.

In each iteration, each ant $\mathbf{x}_i$ selects a road. The probability of choosing the road to the ant $\mathbf{x}_j$ is calculated by

$$p^t(\mathbf{x_i}, \mathbf{x_j}) = \frac{[f^t(\mathbf{x_i}, \mathbf{x_j})]^\alpha \left[\frac{1}{L_{ij}^t}\right]^\beta}{\sum_{\alpha \in N_i^k}\left([f^t(\mathbf{x_i}, \mathbf{x}_\alpha)]^\alpha \left[\frac{1}{L_{i\alpha}^t}\right]\right)}, \tag{4}$$

where N_i^k is a set of unvisited places by ant k but leading to i and α is the impact of left pheromones.

The movement of ant $\mathbf{x}_i$ is based on the selection of path with the best probability to the ant $\mathbf{x}_j$. It may be represented as the following equation

$$\mathbf{x_i^{t+1}} = \mathbf{x_i^t} + \text{sign}(\mathbf{x_i^t}(\text{ind}(t)) - \mathbf{x_i^t}), \tag{5}$$

where ind(t) is an array of neighbor indices after sort. In practice, each ant can move in one of eight directions where the pheromone level is the highest.

B. AACA adapted to generate mazes

In each iteration, the ants move in search of food which represents the exit of the maze. Each ant strives to find a way out, which is created at random on one of the walls in the initial phase of the algorithm. This exit is marked by increasing the value of pheromones. Such adaptation algorithm allows us to create an array of pheromones, which will represent the maze. For each value of the array of pheromones a corresponding function is adopted

$$\Lambda(y) = \begin{cases} y \in \langle 0, \zeta) & \text{empty space} \\ y \in (\zeta, 1\rangle & \text{walls} \end{cases}, \tag{6}$$

where y is the value of the pheromone, ζ is a parameter denoting a minimum value for which formed the wall. In order to create more complicated maze, we can add parameter r which means the number of random alleys. The parameter is the amount of the walls, which also will be removed. For example, for the number 1 in the array of pheromones it means four walls. If the value r is greater than 0, there it is 50% chance that one of the walls does not create. The entire process takes place in a random way depending on the value of the r. Again the value is 0 and the walls are removed but only those that are adjacent to the same value. Visualization of the process is shown in Fig. 1. A complete algorithm is described in Algorithm 2.

C. The march of the Queen

In order to generate a maze, stop condition must be modified. In [18], the authors selected number of iterations, again in [32] accuracy of the obtained solution was chosen as a condition for the end of the algorithm. Modification of stop condition is to obtain road from the entry point to the exit point located on the other side of the maze.

At the end of each iteration, the queen tries to go through the created maze to evaluate the work of its colonies. If the queen finds a way out of the maze, it means that she is satisfied with the work of ants and this part of the algorithm is done. Otherwise, the next iteration of the algorithm starts because the ants did not manage to build a road. While searching for a way out, the queen moves according to the Cartesian metric defined in (3). The queen movement is prevented through the crossing on the diagonal by the following condition

$$L_{ij} = 1. \tag{7}$$

In addition, area with the wall is deleted before choosing it in the march of the queen. The algorithm with a stop condition is shown in Algorithm 1.

Algorithm 1 The march of the Queen

```
 1: Start,
 2: Create an array of pheromones in accordance with (6),
 3: Find all the entrances to the maze,
 4: for each entry to the maze do
 5:    while there is no other movement do
 6:       Find neighboring fields,
 7:       Remove fields in a row, in which the Queen was in
          the previous step,
 8:       for each neighboring fields do
 9:          if equation (7) is not true or the field is a wall then
10:             Delete field,
11:          end if
12:       end for
13:       Select at random one of the existing movements,
14:    end while
15:    if the last field is one of the entrances then
16:       End of the algorithm - there is a way out of the maze,
17:    end if
18: end for
19: End of the algorithm - there is no way out of the maze,
20: Stop.
```

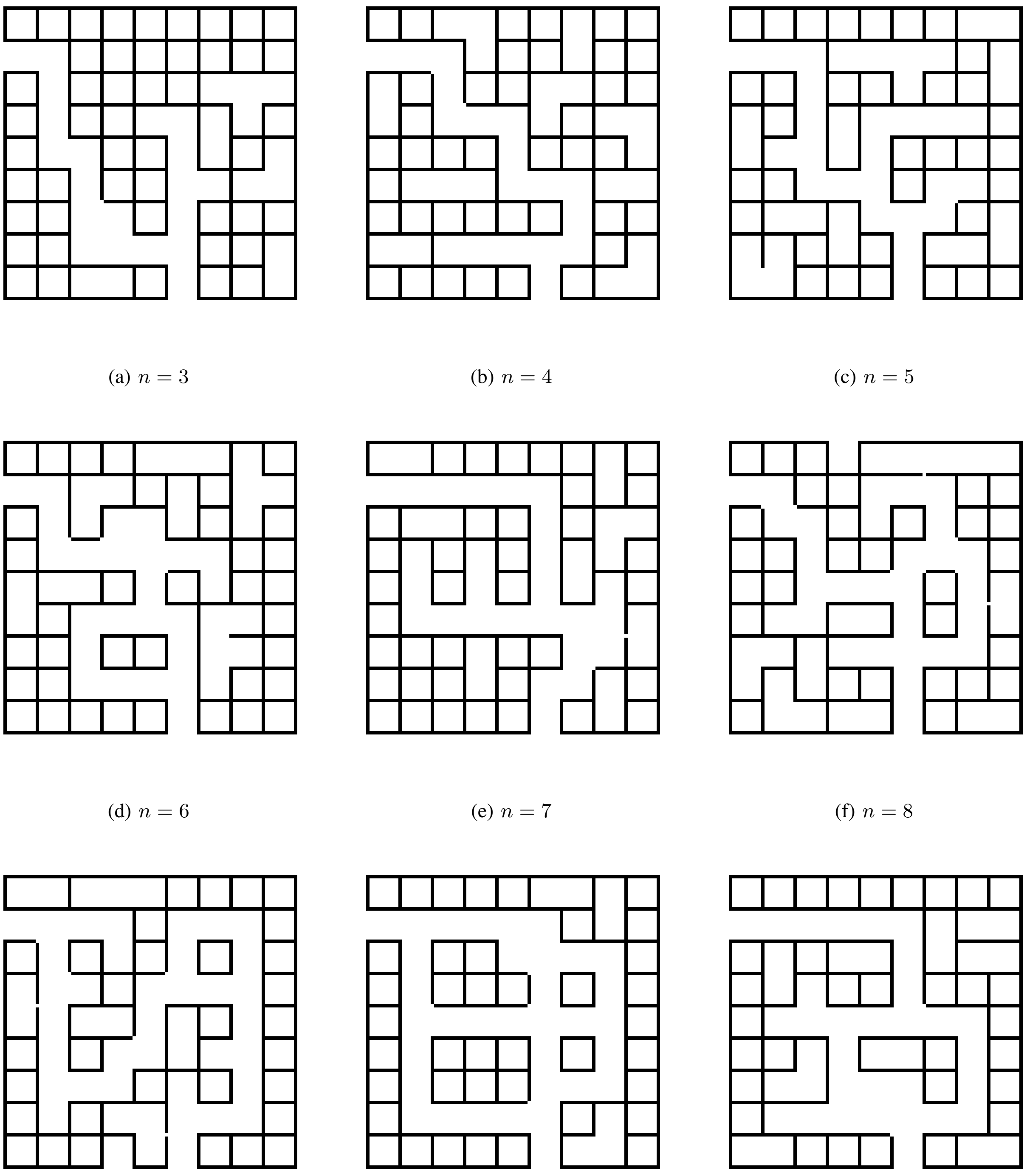
III. Experiments

Tests were performed to create different mazes using AACA. Results were examined in velocity and correctness. Research carried out for mazes are presented in the form of a square (see Fig. 2 and Fig. 5) and rectangular (see Fig. 4), where the following parameters were applied

- for square maze - $\alpha = 0,3$, $\rho = 0.3$, $\zeta = 0.5$, $r = 15$;
- for rectangular maze - $n = 20$, $\alpha = 0,3$, $\rho = 0.3$, $\zeta = 0.5$, $r = 40$.

The Fig. 4 shows generated mazes depending on the number of indiviuals in the population. The results show a correlation between the amount of ants and the quality of maze - the more ants, the more winding roads and consequently maze became too trivial. For this reason, the number of ants should fulfill the following condition

$$n^2 < S, \tag{8}$$


(a) $n = 3$ (b) $n = 4$ (c) $n = 5$

(d) $n = 6$ (e) $n = 7$ (f) $n = 8$

(g) $n = 9$ (h) $n = 10$ (i) $n = 11$

Fig. 2: Generated mazes depending on the number of ants in the population. All mazes are 9×9 and exits are located in the same place for each maze.

where n is the number of ants and S means the value of the area of the maze.

During each test, 1000 measurements were performed. Each of the resulting mazes was different from the other, which leads to the conclusion that uniqueness was obtained. In addition, the measurements of labyrinths create time depending on the value of the field area are considered. Time measurements were averaged using the arithmetic mean and the results are shown in Fig. 3. Based on the chart, creating a large maze does not require a lot of time but the greater area of the maze, the more time is needed to construct.

IV. Conclusions

In the research, tests were performed on multiple mazes in terms of different combinations of inputs. The results showed that Artificial Ant Colony Algorithm can create mazes that may serve as boards for 2D games. Even with large dimensions, the algorithm creates labyrinths in various configurations very quickly, what is the effect of the application of the additional algorithm as a condition for stopping the algorithm. Large randomness of AACA provides an additional advantages, which is uniqueness for the same input data.

This paper presents an alternative method to the existing algorithms ([13], [14], [28], [33]) designed to create mazes of different sizes. In addition to the aforementioned advantages, an important aspect is the quality of generated mazes. Quality can be called the amount of fun while searching for a solution. Unfortunately, it is immeasurable from a mathematical point of view. The quality can also be understood as the size and number of winding roads that make it difficult to find a way out. In this case, the proposed algorithm allows the user to control the quality of the mazes through a number of input parameters.

Algorithm 2 AACA to create mazes

```
 1: Start,
 2: Define all coefficients: $n$ size of workers population, $\alpha$ impact of left pheromones, $\rho$ evaporation rate, $\zeta$ the minimum value of the pheromone, $r$ number of random alleys,
 3: Create array with $\alpha$ values and two different exits,
 4: while the queen does not pass the entire maze do
 5:    Update pheromone values using (1),
 6:    Calculate distances between worker ants (3),
 7:    Calculate possible path to follow by worker $i$ to location $j$ $p^t(\mathbf{x_i}, \mathbf{x_j})$ using (4),
 8:    Determine the best position to follow,
 9:    Move population of workers using (5),
10: end while
11: Calculate the value of pheromone according to (6),
12: Create a bitmap $I$,
13: for each value $v$ of pheromone do
14:    if $v$ is 1 then
15:       if $r > 0$ then
16:          if random value is less than 0.5 then
17:             Create a wall.
18:          end if
19:       else
20:          Create a wall.
21:       end if
22:    else
23:       for each neighbor of $v$ do
24:          if neighbor is 1 then
25:             Create a wall.
26:          end if
27:       end for
28:    end if
29: end for
30: Stop.
```

References

[1] D. Polap, M. Wozniak, C. Napoli, E. Tramontana, and R. Damasevicius, "Is the colony of ants able to recognize graphic objects?" in *Information and Software Technologies*, ser. Communications in Computer and Information Science, G. Dregvaite and R. Damasevicius, Eds. Springer International Publishing, 2015, vol. 538, pp. 376–387.

[2] C. Napoli, G. Pappalardo, E. Tramontana, and G. Zappalà, "A cloud-distributed gpu architecture for pattern identification in segmented detectors big-data surveys," *The Computer Journal*, p. bxu147, 2014, doi: 10.1093/comjnl/bxu147.

[3] C. Napoli, F. Bonanno, and G. Capizzi, "An hybrid neuro-wavelet approach for long-term prediction of solar wind," in *IAU Symposium 274*, 2010, pp. 247–249.

[4] G. Capizzi, F. Bonanno, and C. Napoli, "Recurrent neural network-based control strategy for battery energy storage in generation systems with intermittent renewable energy sources," in *IEEE international conference on clean electrical power (ICCEP)*. IEEE, 2011, pp. 336–340. [Online]. Available: http://dx.doi.org/10.1109/ICCEP.2011.6036300

[5] G. Capizzi, C. Napoli, and L. Paternò, "An innovative hybrid neuro-wavelet method for reconstruction of missing data in astronomical photometric surveys," in *Artificial Intelligence and Soft Computing*. Springer Berlin Heidelberg, 2012, pp. 21–29.

[6] C. Napoli, F. Bonanno, and G. Capizzi, "Exploiting solar wind time series correlation with magnetospheric response by using an hybrid neuro-wavelet approach," in *IAU Symposium 274*, vol. 6. Cambridge University Press, 2010, pp. 156–158.

[7] G. Capizzi, F. Bonanno, and C. Napoli, "A wavelet based prediction of wind and solar energy for long-term simulation of integrated generation systems," in *Power Electronics Electrical Drives Automation and Motion (SPEEDAM), 2010 International Symposium on*. IEEE, 2010, pp. 586–592.

[8] C. Napoli, G. Pappalardo, and E. Tramontana, "A mathematical model for file fragment diffusion and a neural predictor to manage priority queues over bittorrent," *International Journal of Applied Mathematics and Computer Science*, vol. 26, no. 1, 2016.

[9] M.A. Khenissi, F. Essalmi and M. Jemni, "A learning version of Pacman game," *Information and Communication Technology and Accessibility (ICTA), 2013 Fourth International Conference on*, pp. 1–3, 2013.

[10] F. Annaz, "A Mobile Robot Solving a Virtual Maze Environment," *International Journal of Electronics, Computer and Communications Technologies*, vol. 26, no. 2, pp. 1–7, 2012.

[11] L. Ferreira, L. Pereira, C. Toledo, "A multi-population genetic algorithm for procedural generation of levels for platform games," *Proceedings of the 2014 conference companion on Genetic and evolutionary computation companion*, pp. 45–46, 2014.

[12] M. Woźniak, C. Napoli, E. Tramontana, G. Capizzi, G. Lo Sciuto, R. K. Nowicki, and J. T. Starczewski, "A multiscale image compressor with rbfnn and discrete wavelet decomposition," in *IEEE IJCNN 2015 - 2015 IEEE International Joint Conference on Neural Networks, Proceedings*. 12-17 July, Killarney, Ireland: IEEE, 2015, (accepted-in press).

[13] H.L. Lee, C.F. Lee, L.H.Chen, "A perfect maze based steganographic method," *Journal of Systems and Software*, vol. 83, no. 12, pp. 2528–2535, 2010.

[14] N. Niwayama, N. Chen, T. Ogihara, Y. Keneda, "A steganographic method for mazes," *Proc. of Pacific Rim Workshop on Digital Steganography*, 2002.

[15] K. Waledzik and J. Mandziuk, "An automatically generated evaluation function in general game playing," *IEEE Trans. Comput. Intellig. and AI in Games*, vol. 6, no. 3, pp. 258–270, 2014.

[16] M. Woźniak and Z. Marszałek, "An idea to apply firefly algorithm in 2D images key-points search," *Communications in Computer and Information Science - ICIST'2014*, vol. 465, pp. 312–323, 2014.

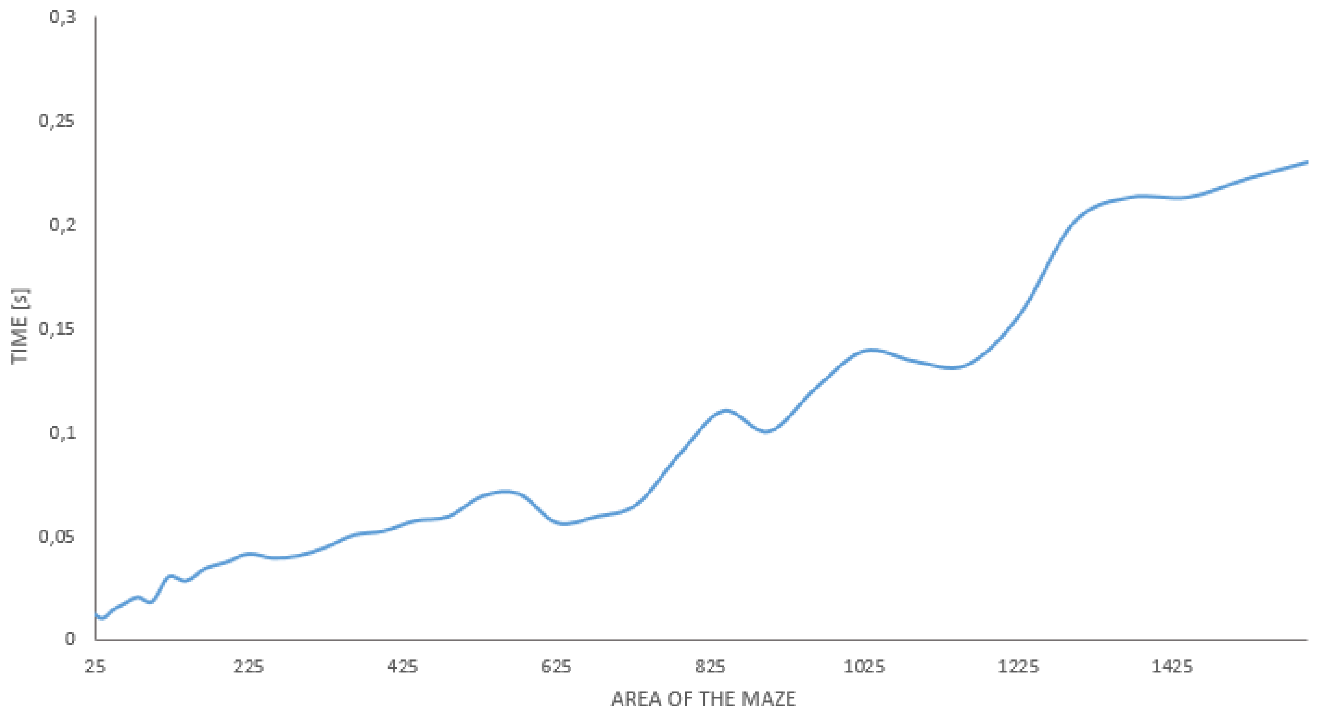


Fig. 3: Time to generate the maze for applied AACA.

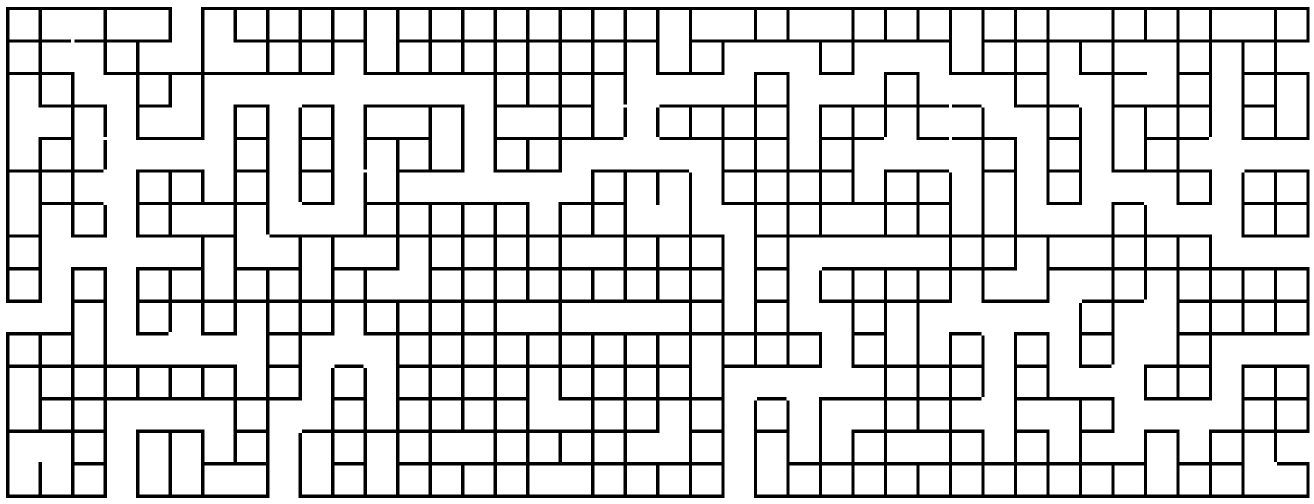


Fig. 4: An example of 15×40 maze generated by the algorithm.

[17] M. Ali, C.W. Ahn, "An optimal image watermarking approach through cuckoo search algorithm in wavelet domain," *Internatinal Journal of System Assurance Engineering and Management*, pp. 1–10, 2014.

[18] T. Liao, K. Socha, M. Montes de Oca, T. Stutzle and M. Dorigo, "Ant colony optimization for mixed-variable optimization problems," *Evolutionary Computation, IEEE Transaction on*, vol. 18, no. 4, pp. 503–518, 2014.

[19] J. Zhang, K. He, J. Zhou, M. Gong, "Ant colony optimization and statistical estimation approach to image edge detection," *2010 6th International Conference on Wireless Communications Networking and Mobile Computing (WiCOM)*, pp. 1–4, 2010.

[20] M. Dorigo, V. Maniezzo and A. Colorni, "Ant system: optimization by a colony of cooperating agents," *Systems, Man, and Cybernetics, Part B: Cybernetics, IEEE Transactions on*, vol. 26, no. 1, pp. 29–41, 1996.

[21] Q. Sun, S. He, "Artificial neural network using the training set of DTS for Pacman game," *Wavelet Active Media Technology and Information*

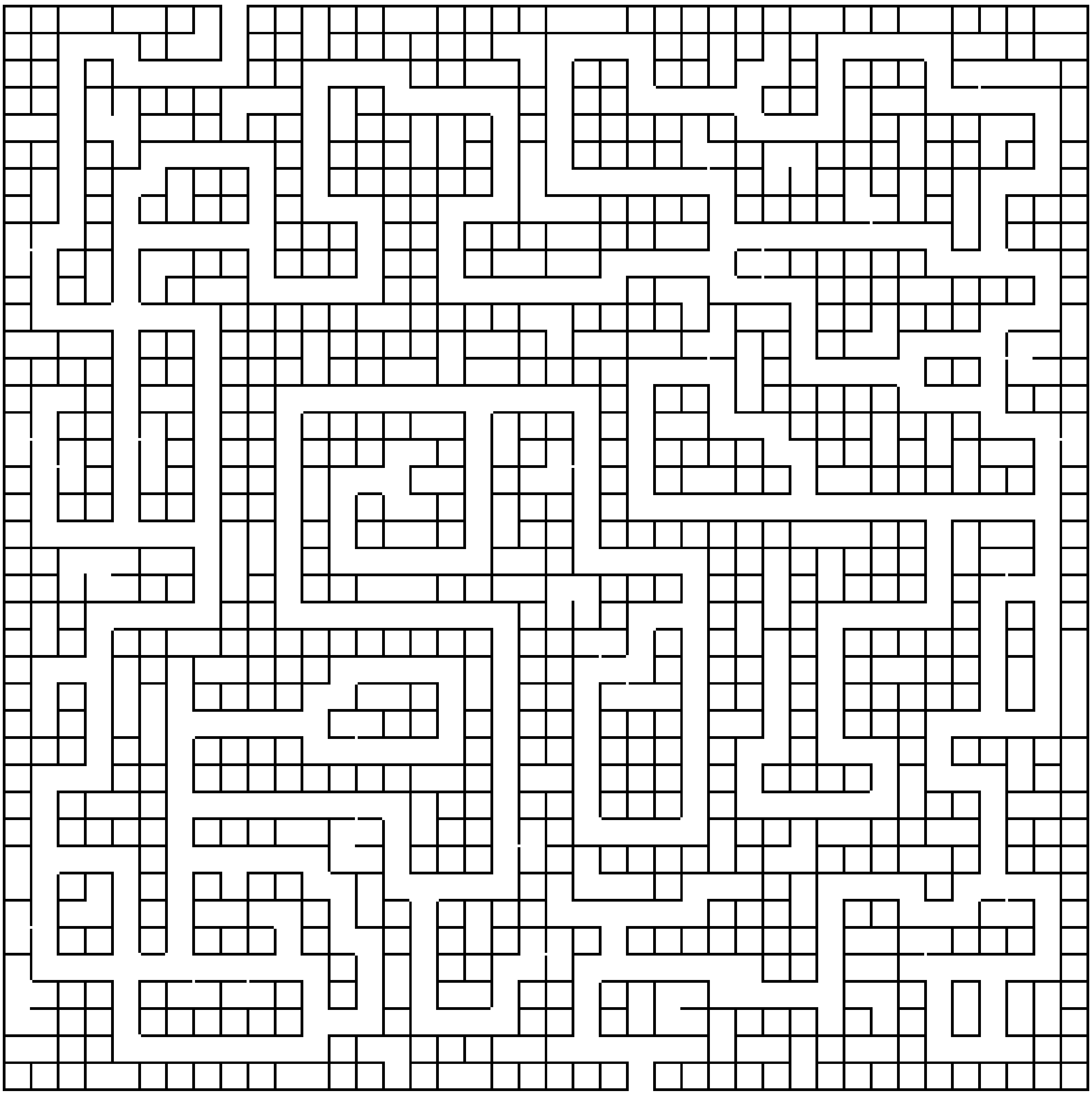


Fig. 5: An example of 40×40 maze generated by the algorithm for $n = 35$.

Processing (ICCWAMTIP), 2014 11th International Computer Conference on, pp. 209–213, 2014.

[22] M. Woźniak and D. Połap, "Basic concept of cuckoo search algorithm for 2D images processing with some research results," in *Proceedings of the 11th International Conference on Signal Processing and Multimedia Applications - SIGMAP'2014*. 28-30 August, Vienna, Austria: SciTePress - INSTICC, 2014, pp. 164–173.

[23] M. Woźniak, D. Połap, M. Gabryel, R. K. Nowicki, C. Napoli, and E. Tramontana, "Can we preprocess 2d images using artificial bee colony?" *Lecture Notes in Artificial Intelligence - ICAISC'2015*, vol. 9119, pp. 660–671, 2015, DOI: 10.1007/978-3-319-19324-3_59.

[24] I. Brajevic, M. Tuba, "Cuckoo search and firefly algorithm applied to multilevel image thresholding," *Cuckoo Search and Firefly Algorithm*, pp. 115–139, 2014.

[25] A. Colorni, M. Dorigo and V. Maniezzo, "Distributed optimization by ant colonies," *Proceedings of the first European conference on artificial life*, vol. 142, pp. 134–142, 1991.

[26] M. Woźniak, "Fitness function for evolutionary computation applied in dynamic object simulation and positioning," in *Proceedings of the Symposium Series on Computational Intelligence - SSCI'2014 : Symposium on Computational Intelligence in Vehicles and Transportation Systems - CIVTS*. 9-12 December, Orlando, Florida, USA: IEEE, 2014, pp.

108–114.

[27] D. Pinelle, N. Wong, T. Stach, "Heuristic evaluation for games: usability principles for video game design,", *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, pp. 1453–1462, 2008.

[28] Y. Okamoto, R. Uehara, "How to make a picturesque maze," in *CCCG*, pp. 137–140, 2009.

[29] J. Xu and C.S. Kaplan, "Image-guided maze construction," *ACM Transactions on Graphics (TOG)*, vol. 26, no. 3, pp. 29, 2007.

[30] M. Woźniak, D. Połap, R. K. Nowicki, C. Napoli, G. Pappalardo, and E. Tramontana, "Novel approach toward medical signals classifier," in *IEEE IJCNN 2015 - 2015 IEEE International Joint Conference on Neural Networks, Proceedings*. 12-17 July, Killarney, Ireland: IEEE, 2015, (accepted-in press).

[31] M. Woźniak, "On applying cuckoo search algorithm to positioning $GI/M/1/N$ finite-buffer queue with a single vacation policy," in *Proceedings of the 12th Mexican International Conference on Artificial Intelligence - MICAI'2013*. 24-30 November, Mexico City, Mexico: IEEE, 2013, pp. 59–64.

[32] M. Woźniak and D. Połap, "On Some Aspects of Genetic and Evolutionary Methods for Optimization Purposes," *Internatinal Journal of Electronics and Telecommunications*, vol. 61, no. 1, pp. 7–16, 2015.

[33] J.B. Kruskal, "On the shortest spanning subtree of a graph and the traveling salesman problem," *Proceedings of the American Mathematical society 7.1*, pp. 48–50, 1956.

[34] A. Kalyanpur, M. Simon, "Pacman using genetic algorithms and neural networks," *University of Maryland*, 2001.

[35] A. Repenning and C. Lewis, "Playing a game: The ecology of designing, building and testing games as educational activities," *World Conference on Educational Multimedia, Hypermedia and Telecommunications*, vol. 2005, no. 1, pp. 4901–4905, 2005.

[36] M. Swiechowski and J. Mandziuk, "Self-adaptation of playing strategies in general game playing," *IEEE Trans. Comput. Intellig. and AI in Games*, vol. 6, no. 4, pp. 367–381, 2014.

[37] M.G. Bellemare, Y. Naddaf, J. Veness, M. Bowling, "The arcade learning environment: An evaluation platform for general agents," *Journal of Artificial Intelligence Research*, vol. 47 pp. 253-279, 2013.

[38] A.B. Lloyd, "The Egyptian Labyrinth," *The Journal of Egyptian Archaeology*, pp. 81–100, 1970.

[39] C. Napoli, G. Pappalardo, E. Tramontana, R. K. Nowicki, J. T. Starczewski, and M. Woźniak, "Toward work groups classification based on probabilistic neural network approach," *Lecture Notes in Artificial Intelligence - ICAISC'2015*, vol. 9119, pp. 76–87, 2015.

[40] H. Desurvire, M. Caplan, J.A. Toth, "Using heuristics to evaluate the playability of games," *CHI'04 extended abstracts on Human factors in computing systems*, pp. 1509–1512, 2004.

[41] M.H. Horng, "Vector quantization using the firefly algorithm for image compression," *Expert Systems with Application*, vol. 29, no. 1, pp. 1078–1091, 2012.

[42] H.N. Teodorescu, M. Rusu, "Yet Another Method for Image Segmentation based on Histograms and Heuristics," *Computer Science*, vo. 20, no. 2, pp. 163–177, 2012.