FOUNDATIONS OF COMPUTING AND DECISION SCIENCES
Vol. 43 (2018) No. 3
ISSN 0867-6356
DOI: 10.1515/fcds-2018-0012 e-ISSN 2300-3405

sciendo

Modeling Biological Systems Using Crowdsourcing

Szymon Wasik *

Abstract. Crowdsourcing is a very effective technique for outsourcing work to a vast network usually comprising anonymous people. In this study, we review the application of crowdsourcing to modeling systems originating from systems biology. We consider a variety of verified approaches, including well-known projects such as EyeWire, FoldIt, and DREAM Challenges, as well as novel projects conducted at the European Center for Bioinformatics and Genomics. The latter projects utilized crowdsourced serious games to design models of dynamic biological systems, and it was demonstrated that these models could be used successfully to involve players without domain knowledge. We conclude the review of these systems by providing 10 guidelines to facilitate the efficient use of crowdsourcing.

Keywords: Computer game, Crowdsourcing, Dynamic system, Modeling, Serious game, Systems biology

1. Introduction

The term *crowdsourcing* was introduced in 2006 by Jeff Howe [32]. However, the concept of crowdsourcing, which is understood as outsourcing work to a vast network of anonymous people in the form of an open call, is much older. It has been discovered that one of its first applications was a method for measuring the longitude of a ship in 1714, for which the British government offered a prize [13]. Subsequently, researchers and entrepreneurs have utilized the concept of crowdsourcing many times, but its rapid progress started after the development of the Internet in the 1990s. Successful examples of its application include services such as Wikipedia and OpenStreetMap. An extended review of crowdsourcing systems available on the world wide web (WWW) was provided in a survey by Doan et al. [17], and a more detailed

*Institute of Computing Science, Poznan University of Technology, Poznan, Poland; Institute of Bioorganic Chemistry, Polish Academy of Sciences, Poznan, Poland; European Center for Bioinformatics and Genomics, Poznan University of Technology, Poznan, Poland; szymon.wasik@cs.put.poznan.pl

discussion regarding the nature of crowdsourcing was given by Estelles and Gonzalez [19]. Crowdsourcing can bring many benefits. As observed by Francis Galton in 1907, the collective opinion of a crowd of individuals can be much more precise than the opinion of any single individual in the crowd. This is an underlying principle of the so-called *wisdom of the crowd* [23], which later evolved into the idea of collective intelligence (CI) [7]. The term CI is a more general concept that is usually defined as an intelligence emerging from the collaboration, collective efforts, and competition of many individuals [58, 52]. In recent years, many platforms have been implemented to support crowdsourcing, such as InnoCentive and CrowdAnalytix [29].

Crowdsourcing has been utilized successfully in many scientific and industry inspired areas. For example, crowdsourcing has been successfully applied in computing science for collecting and processing data in projects such as Wikipedia [38, 57, 46], reCaptcha [63], and OpenStreetMap [25], and many other areas including data mining [73] and natural language processing [28]. Moreover, some platforms were designed primarily to support crowdsourcing for tasks (micro-tasking platforms), with Amazon Mechanical Turk [36] being the best-known example. This system allows requesters to define simple tasks that can be performed easily by humans (such as transcribing audio recordings), and matches them with crowdsourced workers. These workers receive a payment for completed tasks. Other services that work on a similar basis include Figure Eight (formerly Crowdflower [62]) and Prolific [48]. There are also platforms that focus on more complicated projects by allowing employers to find employees to complete time-consuming tasks, such as ClickWorker and Upwork [14].

Another fascinating application of the crowdsourcing concept is specialized computer games where players can have fun and solve complex scientific problems at the same time [55]. These games are called *crowdsourced serious games* (this term was first introduced by Tellioglu et al. in 2014 [59]) and they have been employed successfully for solving problems that originate from various fields of science. One of the first crowdsourced serious games was FoldIt, which allows players to fold a protein to obtain its tertiary structure. The results obtained by the players were so good that all of the participants were recognized as co-authors on the paper published in *Nature* [10]. Other biological applications of this concept include EteRNA [37], EyeWire [26, 3], Phylo [33], Stall Catcher [9], Nanocrafter [4], MalariaSpot [41], and BioGames [43]. In addition, significant applications of crowdsourced serious games have been made in fields other than biology, such as the applications developed by Verigames.com for the formal verification of software [40] and the *Throw the Hamster* game designed for deducing mathematical formulae from data [68].

Finally, another innovative implementation of the crowdsourcing technique is in online judge systems [67, 66]. These online platforms allow the submission of algorithm implementations (or their output data) to the cloud then they are evaluated in a homogeneous, reliable environment. These systems can be used to rapidly organize crowdsourced challenges where participants submit algorithms that an online judge automatically collects, evaluates, and compares. Many of these systems have been used to solve scientific problems, support teaching, support recruitment, or even to develop software components in the cloud. A detailed review of these systems was provided by Wasik et al. [67].

Many of the applications listed above are related to *systems biology*, which is a relatively new field of biology defined as the antithesis of the approach that was applied previously for centuries. Previously, the development of biology was driven by a reductionist approach, where in order to understand the functioning of an organism as a whole, it is necessary to distinguish the fundamental life processes that occur in them and understand each of them separately [47]. Researchers were convinced that if we could describe each process accurately, then complete knowledge would be available to understand the operation of the whole organism directly. This approach drove the continuous development of observational methods that allowed observations of organisms at an ever-increasing level of detail. This ranged from observing the operation of whole organs, through tissues, to individual cells and the processes occurring in them, and even biochemistry and biophysics at the levels of individual particles and atoms. Finally, this process yielded precise descriptions of hundreds of processes that occur in biological organisms, without providing answers to many vital questions. A good example is a neuron in the brain. The transmission of electrical impulses that allow neuronal communication and information transmission is very well understood, but it does not explain how thoughts arise and how the process of thinking proceeds [21].

The problem introduced above was first addressed in the second half of the twentieth century when it was noted that a more global view is necessary in addition to analyzing particular processes. This gave rise to a new field of biology called *systems biology*, which involves studying the complex interactions occurring in biological systems. One of the primary goals of systems biology is to analyze what was lacking in the reductionist approach. This involves analyzing properties that only emerge in a global context (so-called emergent properties). Therefore, systems biology can be considered the most important representative example of a holistic approach to biology [35].

Formally, the term systems biology was introduced in 1966 by Mihajlo Mesarovic through the organization of an international symposium on *Systems Theory and Biology* [44]. However, the first research in this field occurred much earlier with a model of the propagation of signals along the axons of nerve cells, which combined knowledge of the behavior of the sodium and potassium molecules found in neurons to observe a more global impulse transmission process [30]. Subsequently, systems biology has been systematically developed, with interest increasing since the 1990s. This was due to two factors: the availability of large volumes of data in the area of functional genomics, and the emergence of more efficient computers. This progress was accompanied by the rapid development and improvement of computers, which facilitated simulations and the analysis of the models. One of the most significant achievements during this period was a model of metabolism in an entire hypothetical cell in 1997 [60]. Currently, systems biology is a highly interdisciplinary field, which combines research in fields such as biology, mathematics, computer science, chemistry, biochemistry, physics, biophysics, and even psychology and sociology. Research conducted in the area of systems biology is used in diverse fields such as genomics, transcriptomics, proteomics, and metabolomics.

In this review, we focus on the principal task related to systems biology; modeling

biological systems [51, 70] using crowdsourcing. Modeling these systems has been addressed in many previous books and articles, such as [16, 1, 56]. However, an extensive overview of the application of crowdsourcing to modeling these systems has not been conducted, although Good and Su gave a comprehensive review of the application of crowdsourcing in bioinformatics [24]. Thus, we first survey the applications of crowdsourcing to modeling biological systems that originate from systems biology. We then present the results of several projects conducted at the Poznan University of Technology related to the modeling of dynamic, biological systems using crowdsourced serious games by focusing on the different aspects of crowdsourcing. The objective of all these projects was to design a game that could exploit the collaborative effort of many players, who lacked any knowledge of dynamic systems modeling, to design a model. Achieving these objectives involved designing a game that concealed all expert knowledge from the users, and exploiting the collaborative effort of the players. Finally, we suggest some guidelines on how to make effective use of crowdsourcing to achieve useful results.

2. Related work

Modeling biological systems is still an emerging application of crowdsourcing, but several platforms have been successfully applied using this technique to obtain valuable biological insights. In the following, we review three widely used platforms: EyeWire, FoldIt, and DREAM Challenges. We also provide a brief review of the application of crowdsourcing to the annotation of systems biology data.

2.1. EyeWire

The objective of EyeWire is to reconstruct the three-dimensional (3D) structure of neurons and their mutual connections. To achieve this goal, players are asked to color fragments of neurons that can be identified inside small cubes retrieved using serial block-face electron microscopy. The initial solutions are constructed using machine-learning based volume segmentation, and the solutions are then improved continuously based on the inputs of the players. Players can investigate the cube using different cross sections, and add new parts of neurons to the initial solution. Technically, addition is achieved by clicking with a computer mouse on the subarea that the player wants to include. To make the process faster, the flood fill algorithm is used to extend this subarea to other areas wherever the algorithm is sure that the neuron continues. The game presents 2D and 3D visualizations of the solution proposed by the player to help them verify correctness. The EyeWire development team organizes special events and contests for motivating players to spend more time solving game tasks. Moreover, players can check daily, weekly, and monthly rank lists, receive badges and achievements, and connect with other players using social networking features. Data collected from the crowd are then used to model dendrite connections in a system of starburst amacrine cells [34].

Since the game was publicly published in 2012, it has helped to achieve many significant results in both biomedical and computational areas. For example, in 2013, the authors created a dense reconstruction of 950 neurons in the inner plexiform layer of a mouse, and they identified the motifs responsible for detecting localized motion [26]. Another important achievement was the publication of a digital library that provides a 3D interactive view of 400 ganglion cells from a single patch of mouse retina and graphs of their visual responses [3].

2.2. FoldIt

An important area of systems biology is interactomics, which focuses on the interactions between molecules. One common applications is the analysis of protein–protein interactions [8], which can be analyzed experimentally or computationally [61]. In the latter method, the most important input data are the tertiary structures of proteins. Thus, the FoldIt game is one of the oldest and best-known crowdsourced games applied in bioinformatics.

FoldIt was first published in 2008, and as of July 2018 it had over 700,000 registered users. The objective of this game is to predict the tertiary structure of proteins with the help of players. FoldIt presents proteins as 3D visualizations, and it ensures that the distances between the atoms in the backbone are correct. The task of the user involves changing the positions of the atoms to minimize the internal energy of the molecule. Moreover, the user is provided with a set of tools to facilitate and accelerate their work. Some incorrect structures are detected and marked automatically. In addition, the player has the option of optimizing the designed structure by executing an automatic parameter tuning procedure.

The highest ranked solutions designed by players are then checked and examined by scientists to determine whether they can exist in reality. This approach has been shown to be highly effective. For example, in 2012, FoldIt organized a challenge to support the drug design process. The organizers asked players to remodel the backbone of a computationally designed molecule to allow additional interactions with substrates, which led to the discovery of new and effective interaction structures [18]. Currently, the authors of the platform are trying to broaden its application range and to adapt it for determining crystal structures. Accordingly, they organized a crystallographic model-building challenge where they invited trained crystallographers, undergraduate students, FoldIt players, and autonomous algorithms. The winner of the challenge was a group of FoldIt players, thereby proving the usefulness of the platform [31].

2.3. DREAM Challenges

Dialogue for Reverse Engineering Assessment of Methods (DREAM) Challenges is an online judge platform that is similar to Kaggle, and it is a crowdsourcing website dedicated to solving science data problems. The main difference is that DREAM

Challenges focuses on problems originating from systems biology and translational medicine [53, 11]. This platform allows the publication of biomedical challenges that can then be solved by scientists. It also provides an intuitive interface for collecting solutions generated by researchers and the implementation of scoring methods. DREAM Challenges hosts challenges using the Synapse platform [15] developed and maintained by Sage Bionetworks, and this collaborative research platform helps research teams to share data, track analyses, and collaborate remotely.

DREAM Challenges have organized 50 successful challenges with a focus on the field of biomedicine. In particular, some were related to systems biology, and one of the most successful was the HPN-DREAM network inference challenge, where the objective was to learn causal influences in signaling networks [42, 27]. The organizers managed to collect and score over 2000 networks from the challenge participants, and many of them were highly efficient at modeling the signaling network.

Another successful challenge focused on network topology identification and parameter estimation for gene regulatory networks [45]. This challenge comprised two sub-tasks, where the objective of the first was to estimate the parameters of a nine-gene regulatory network, and the participants also had to identify three missing links in an 11-gene network in the second sub-task. By analyzing the results of the challenge, the organizers were able to identify new efficient variants of methods for modeling gene regulatory network. Moreover, they were able to combine components from the various models submitted by participants to construct a new solution that scored better than the winner of the challenge, thereby demonstrating that it is possible to make use of the wisdom of the crowd to construct biological models.

2.4. Text annotation

Research related to systems biology primarily involves the analysis of biological networks such as protein–protein interaction networks, gene regulatory networks, and metabolic networks. However, constructing these networks is a laborious process, where the biggest problem is identifying edges that exist in the network. These edges usually model interactions that have been observed in real biological systems, and information about them can be found in various scientific studies. Unfortunately, each edge is usually described in different studies, and identifying edges requires an extensive literature review. Fortunately, this review process can be distributed among many researchers who work in parallel. Thus, researchers have tried to crowdsource this task to the academic community.

The best example of the utilization of a crowd to annotate biological network is the study by Wang et al. [64]. In this project, a crowd of 70 researchers annotated and analyzed over 4000 gene expression profiles available from the NCBI Gene Expression Omnibus [5]. Consequently, Ansari et al. proposed a tool for verifying biological networks. This tool visualizes networks and allows crowdsourced researchers to verify the existence of interactions based on reported evidence [2]. Finally, WikiPathways facilitates the sharing and editing of information about biological pathways using the Wikipedia platform [49].

3. Materials and Methods

We conducted three projects at the Poznan University of Technology. The objective of the first project, called *Throw the Hamster*, was to verify the possibility of using crowdsourcing games to model dynamic systems. Such systems are very often analyzed by researchers in the area of systems biology as they can model how the system consisting of some biological entities changes in time. Using such a model they can analyze various properties of the model. For example, during all projects, we tried to construct a model of viral infection that can be used to support designing new vaccines and treatment methods [69]. The project was very successful and the results were published in 2015 [68], as summarized briefly in Section 3.1. The goal of the second project was to improve the original approach by solving some problems identified during the verification phase. Finally, we conducted the third project, called *Race the Hamster*, to develop the most efficient method for integrating crowdsourced serious games with micro-tasking services. In the following, we briefly discuss the results of the latter two projects by focusing on aspects related to crowdsourcing.

During each project, the goal of the users was to construct a formula for a function to characterize a dynamic system, i.e., a system such that for each time point $t \in T$, each point of the phase space $m \in M$ is mapped onto another point $m' \in M$. This system can be described by a function:

$$T \times M \to M, \tag{1}$$

and it is usually defined by a differential equation or a system of equations that model the changes in the values of points m in time, i.e.:

$$\dot{m} = f(m). \tag{2}$$

We did not model the system directly by designing a system of differential equations, which is a highly difficult task, but instead we tried to construct a single function to give the values of points m in time, i.e., the solution of a system of differential equations for some initial conditions m_0:

$$m = f(t). \tag{3}$$

To evaluate each solution f, we used a standard mean absolute error defined using the following formula:

$$s(f) = \frac{1}{N} \sum_{i=1}^{N} |m_i - f(t_i)|, \tag{4}$$

where N denotes the number of points in a test dataset T, m_i denotes the value of the test data point collected at time t_i, and $f(t_i)$ is the value estimated by the player's solution. The test dataset T contained data regarding hepatitis C virus infections provided by Dahari et al. [12, 69], which described how the viral RNA level decreased during therapy with PEGylated interferon alpha and ribavirin.

As the solution becomes more accurate, the value of the scoring function $s(f)$ decreases and extremely small values close to 0 are obtained for very good solutions.

This behavior is not intuitive for users without a mathematical background, who generally consider that it is better to have more points in the game. To make the results more user friendly, we defined a transformation of the function $s(f)$, which always returns a positive number that is less than or equal to $10,000$, as follows:

$$p(f) = \left\lfloor \frac{10000}{max(\sqrt{s(f)}, 1)} \right\rfloor \tag{5}$$

To verify how the results obtained using crowdsourcing compared to the results produced by a state-of-the-art artificial intelligence algorithm, we compared the results from the *Throw the Hamster* game to the results generated using symbolic regression implemented in Eureqa software [54]. The comparison demonstrated that the crowdsourcing approach could compete with the machine learning procedures (for further details, see [68]). An essential step when assessing these two approaches was comparing the complexity of the functions. The complexity of a model is usually correlated with the accuracy that it can achieve, so we analyzed the complexity of the models generated by both the Eureqa software and the crowd. This allowed us to avoid over-fitting and generating excessively complex models. We used a method for calculating the complexity of formulae implemented in the Eureqa, which assigns a weight to each of the elementary operations that can be used to construct formula and then sums the weights of all operations. The same procedure was used to compare the results obtained by the *Throw the Hamster* and *Race the Hamster* games. The weights used by this method are presented in Table 1.

Table 1. Weights for operations used to measure the complexity of the solution.

Operation	Weight	Operation	Weight
addition	1	negation	1
subtraction	1	exponentiation	5
multiplication	1	logarithm	4
sine	3	natural logarithm	4
cosine	3	division	2
tangent	4	constant	1
cotangent	4	variable	1

3.1. Original approach: Throw the Hamster game

The original approach was implemented as an Internet application in the form of a game accessed via a web browser [68]. This game provided a platform that users could employ to design and share their proposed model and improve existing ones. The objective of the game was to design a spaceship for a hamster. The flight path of the spaceship was predetermined by the function designed by the player by adding improvements to the technological tree. Each improvement was then mapped to some

mathematical operator. The value of the function's argument modeled time, and it corresponded to the horizontal position of the hamster. The value of the function corresponded to the height at which the hamster was flying at a specific point in time. The goal of the game was to develop a trajectory that matched (as closely as possible) with the individual points from the test dataset T. The visualization phase, for visualizing the designed function in the form of a flight, allowed the user to check the quality of this function by observing how closely the hamster's flight matched with the stars representing the test dataset T. Finally, the quality was presented in the form of a score calculated according to Equation 5. The user could then make further corrections to his solution to make it fit closer to data points. The player could also display the ranking of some of the best solutions posted by other players then load them to continue improvement on these models. Thus, the players could work together to build the best solution and take advantage of the wisdom of the crowd.

The *Throw the Hamster* game was a great success. We found that by achieving much lower complexity, the user-constructed functions were only slightly worse than the solutions generated by Eureqa. In addition, many possible improvements were observed, which motivated us to continue working on the next version of the game. Based on the surveys conducted at the end of the experiment, it appeared that most of the game players did not enjoy the game very much, which greatly hindered the achievement of the intended purpose. We also found that users were concerned by the fact that they did not understand the connection between the function building stage and the flight path in the visualization. This design was due to the assumed need to hide any elements that required mathematical knowledge from the players, but it was only one reason for their concern. Finally, we observed that most of the time spent by users during game playing involved tuning the parameter values in the model, which is a task that could be easily automated.

3.2. Redesigned approach: Race the Hamster game

Based on the conclusions described in the previous section, we redesigned the game and implemented a new version called *Race the Hamster*. In the following, we describe the changes made to the game.

Change #1 The first change was a modification to the game play process. To make the game more interesting for the players, the new version involved driving a race car on a track (see Figure 1). This change introduced additional interactivity into the game because the player must complete the track as quickly as possible by steering the car with the arrow keys on the keyboard. At the end of the race, the player received points that are awarded based on both the quality of the designed function and the time needed to complete the entire track. The shape of the track itself was defined by the function that the player designed, and this function was visualized in a polar coordinates system. Therefore, the time corresponded to the angular coordinate and the modeled value corresponded to the radial coordinate. To ensure that the track

formed a loop (the beginning met with the end), we used Hermite interpolation to generate a segment of track that connected the beginning and the end. The game visualized points from the test dataset T above the track as boosters. These points gave the car additional acceleration to allow the race to be completed in a shorter time. Because of this modification, the player was interested in designing a function that fitted the points as closely as possible.

The adjustment to the game play process required a change to the formula presented in Equation 5, which defines how the scores are awarded to the player. The redesigned formula is defined as follows:

$$p'(f) = w_1 \cdot p(f) + 10000 \cdot w_2 \cdot \max\left(\frac{k \cdot t_{min} - t_p}{(k-1) \cdot t_{min}}, 0\right), \tag{6}$$

where t_p denotes the time that a player requires to complete the track, t_{min} is the minimal time required to complete the track, and k controls the difficulty (it is easier to obtain more points when k is higher). The most important problem is how to select the weights w_1 and w_2. Increasing w_1 raises the importance of the quality of designed function. Increasing w_2 gives the player more fun during the racing phase because the completion time is more important. After studying the players' opinions, we decided to use $w_1 = 0.25$ and $w_2 = 0.75$.

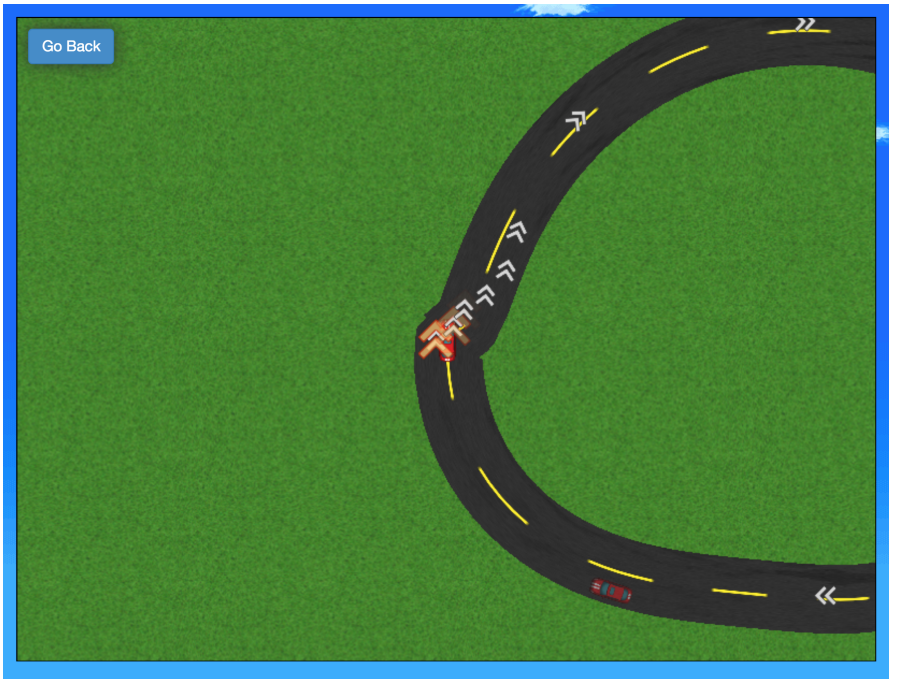

Figure 1. Part of the racing track generated based on the function presented in Figure 2. The boosters that speed up the racing car are closer to the center of the track when the function is closer to the data points.

Change #2 In both versions of the game, the players constructed functions to describe the modeled system by constructing its tree representation. However, the original approach provided a non-intuitive interface for adding and removing new nodes. In the new version of the game, the interface was changed to drag and drop. An example screenshot showing a tree representation of a function is presented in Figure 2.

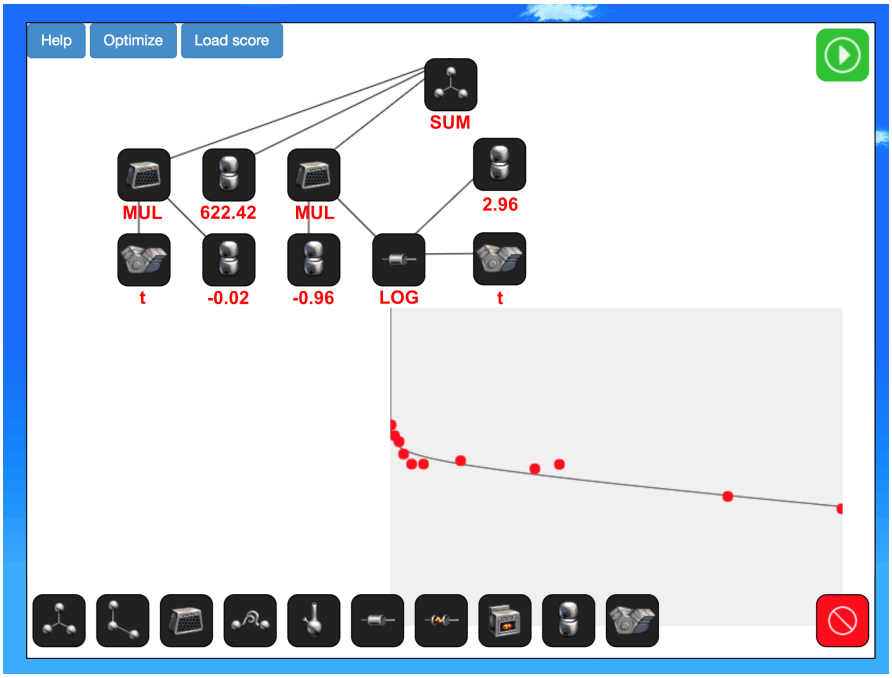

Figure 2. Tree representation function: $f(t) = 622.42 - 0.02 \cdot t - 0.96 \cdot \log_{2.96} t$. The red labels are not visible during the game, and they have been added to show how the tree is translated into the formula. A visualization of the function represented is shown below the tree.

Change #3 In the original version of the game, modifying the modeled function required changes to the structure of the tree before executing the visualization procedure to evaluate the quality of the newly created solution. This was a major problem because each iteration of changes required launching an animation that lasted several additional seconds. The solution to this problem involved previewing the function continually during its creation. The function was visualized in the design stage by presenting a graph in Cartesian coordinates based on a line plotted on a plane containing visualized points from the test dataset T (see Figure 2). Thus, the players could immediately assess any positive or negative consequences of the change, thereby allowing them to notice the relationships between the individual elements that modeled the appropriate mathematical operations.

Change #4 During the design of a function to describe the model, the elements that occurred in each tree were constants. In the first version of the game, the numerical values of these constants were determined by the player. This requirement was very tedious because the player had to find the right structure of the whole tree and optimize all of its constant values manually. Thus, most of their time spent designing functions involved verifying how small changes in the values of the parameters in the model affected the outcome of the game. The problem of tuning the parameter values in the model could easily be automated using computational methods, thereby allowing the player to avoid performing the longest and most boring part of the solution design process. Therefore, this automated step was integrated into the redesigned version of the game. The details of the algorithms employed are described in the next section.

3.3. Automated parameter tuning

The problem of finding values for constants that maximize the score of a player is a well-known nonlinear regression problem. The only requirement is that it should be as computationally efficient as possible to return the results to the user in an acceptably short time. Moreover, because the player had full freedom when designing the function, we assumed that a derivative-free method was required. Thus, we used a constrained optimization by the linear approximation (COBYLA) algorithm ([50]), which is currently the most efficient method (good results are obtained in a relatively short time [22]). This numerical optimization method is applicable when the derivative of the objective function is not known, and it works by iteratively approximating the constrained numerical optimization problem with linear programming problems.

3.4. Crowdsourced parameter tuning

In the tests with the COBYLA algorithm, we found that the method for estimating the values of the real parameters could be improved, and it remains an active research topic [39]. For example, according to our preliminary tests, we observed that it was possible to achieve much better results in only a slightly longer time by designing a hybrid method that uses COBYLA as a local optimization method and an evolutionary approach to escape from local optima. Moreover, the problem of estimating real parameter values formulated as a global optimization problem can be applied in many other areas of systems biology (e.g., [74, 70]). Therefore, we decided to use crowdsourcing to help solve this problem, which we implemented with the Trochilus platform.

The objective of developing the Trochilus platform was to implement a black-box evaluation of optimization algorithms. The architecture was inspired by the software supplementing special session on real parameter optimization organized annually during the IEEE CEC conference. During the session, the organizers announced the results of the challenge, where the task was to design the best-performing method for

the optimization of real parameters. The original aim of the challenge was that the evaluation software should conduct black-box assessments. Therefore, the algorithm used to search for the parameter values did not know the optimized function, and it was only allowed to query the values for some set of parameter values by sending requests to the evaluation software. The algorithms were then evaluated based on the number of requests required to find a solution.

There were two primary problems related to the CEC session: the evaluation process was performed off-line only once each year, and the functions employed were artificial functions with no relationships to any biological problem or other practical problem. Thus, we implemented a crowdsourcing platform with an online judge architecture, which allowed the collection of algorithms to perform real parameter optimization and their comparison. An example screenshot of the platform is presented in Figure 3. The resulting Trochilus platform was presented at several conferences. The feedback from the community was so encouraging that it was expanded into a general purpose crowdsourcing platform for evaluating algorithms to solve any optimization problems. The target platform called Optil.io has already been used to support several international programming challenges [66, 65].

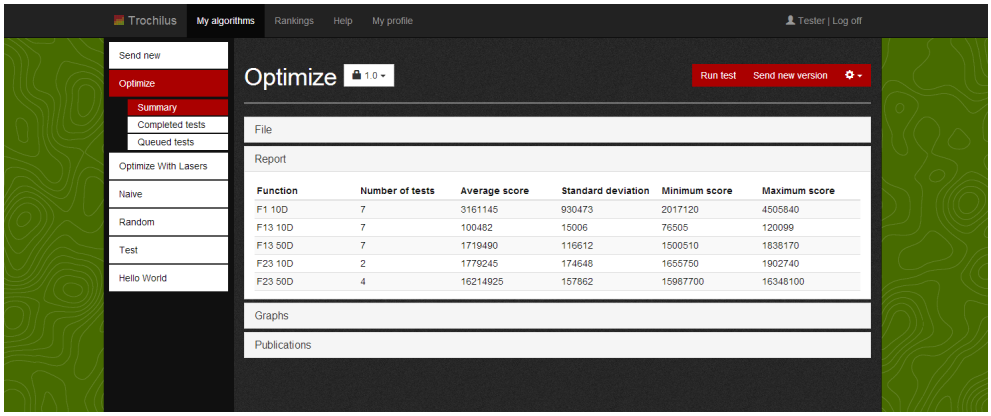

Figure 3. Screenshot of the Trochilus platform showing the results of tests with submitted algorithms using five functions from the CEC 2013 challenge [39].

3.5. Integration of a crowdsourced game with micro-tasking services

Having fun playing a game is often sufficient motivation to encourage players to spend time playing a crowdsourced serious game. However, it is much easier to reach crowdsourced users if they are offered additional payments for completing crowdsourced jobs. Thus, we investigated the possibility of integration with existing micro-tasking services. The best possibility that offered application programming interfaces (APIs) for integration with external services was Figure Eight. Unfortunately, playing the

game required a sophisticated application that could not be integrated internally with Figure Eight using the limited features provided on this platform. Thus, we used a tokens mechanism that is available on the platform for integration with tasks that are solved externally. The integration procedure is explained as follows.

1. The user navigated to the Figure Eight platform and found a page describing the crowdsourced serious game.
2. The user was redirected to the game from the information page.
3. The user played the *Race the Hamster* game and the game generated a token if the result was sufficiently high.
4. The user copied the token to the clipboard and was then redirected to the Figure Eight website.
5. The user pasted the token into a dedicated text area.
6. Figure Eight sent a request to the game to verify whether the token was valid and that it had not been used previously.
7. After positive confirmation, the user was paid for the task.

Unfortunately, the Figure Eight API was too limited to automate the process required to send the token from the game to the platform.

4. Results

4.1. Testing procedure

We conducted three phases of tests: two used two versions of the *Throw the Hamster* game (for details, see [68]), and the third employed the *Race the Hamster* game. We collected no personal data, and players could provide a username to allow them to correlate their consecutive attempts. However, this was not obligatory and we did not check the uniqueness of different attempts. The numbers of players and games played during each phase are presented in Table 2. Over 14,000 games were played, and this allowed us to verify our approach in detail. The average number of games played by each player in the *Race the Hamster* game was almost four times larger than that in the previous version of the game (65.96 vs. 16.94). This was attributable to the more exciting game play and the elimination of the tiresome process required for setting the parameter values.

4.2. Results and analysis

Figure 4 compares the results obtained with the *Throw the Hamster* and *Race the Hamster* games. These results show that the improved version of the game generated significantly better results for most of the solution complexities. The original

Table 2. Number of games played during the first and second iteration of tests with the *Throw the Hamster* game (#1 and #2) and the *Race the Hamster* game (#3).

Iteration	Players	Sessions	Games	Games per player
#1	616	928	7628	12.38
#2	90	212	1525	16.94
#3	85	149	5607	65.96
Total	**791**	**1289**	**14760**	**18.65**

approach achieved slightly better results for complexities in the range from 14 to 17. Moreover, it was possible to overcome the problem of poor quality solutions with high complexities. After the automatic parameter tuning method was introduced, it was much easier for the users to handle complex models. Furthermore, they were able to make improvements, even for the model with a relatively high complexity (59). In the tests of the original approach, the most complex model where the value of the objective function increased had a complexity of 14, which probably also explains why the original approach was better for the low range of complexities. The players found it difficult to design more complex functions. Therefore, they focused on those with lower complexity, but a much smaller subset of the solution space was searched.

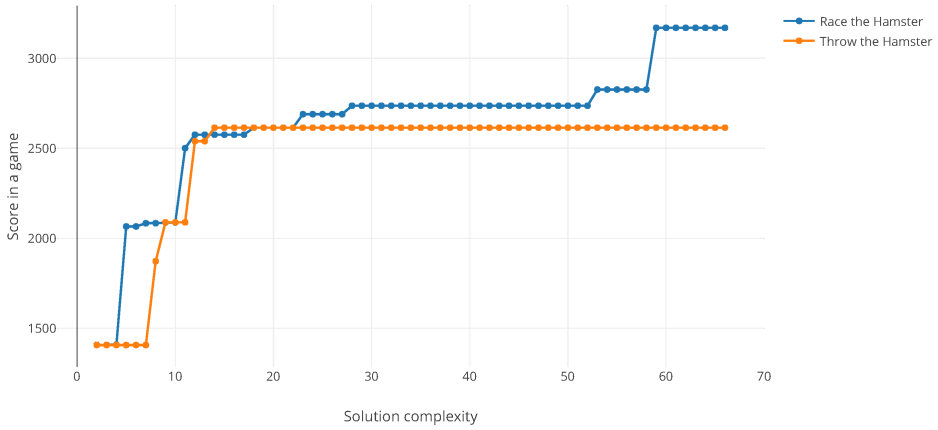

Figure 4. Scores for the best solutions generated by players where the complexity was less than or equal to a specific value.

The functions designed by the players are illustrated as follows. All of the functions presented in the following have a complexity of 23, which was the first complexity

at which the improved approach was always better. Equation 7 is the best function designed using the *Throw the Hamster* game where the complexity is 23 and it scored 2614 points. Equation 8 is the best function with the same complexity, but it was designed using the *Race the Hamster* game and it scored 2689 points. Equations are not simplified because they are presented in a precisely same form as the player designed them.

$$f(t) = 649.23 + \left(-1.019 \cdot 20 \frac{\log(1+t)}{\log(2.715)}\right) + (-0.022 \cdot t) + 1.13 + t \cdot t \cdot 0.0001 \cdot 0.0001 \cdot 0.2 \tag{7}$$

$$f(t) = 649.23 - 1.03 \cdot \log_{2.67}(t) - 0.01 \cdot t + t \cdot t \cdot -0.01 \cdot 0.02 \cdot 0.01 \tag{8}$$

Finally, in both projects we tried to collect the opinions of the players regarding the games. Our analysis of the collected opinions showed that the changes introduced in Race the Hamster game were rated positively. On average, more people stated that they liked the game, and that they would recommend it to a friend as an interesting scientific contribution. In addition, fewer people stated that the game involved an excessive amount of mathematical thinking.

5. Conclusions

The examples presented in this review demonstrate that crowdsourcing can have diverse applications to support the modeling of biological systems. First, crowdsourced serious games can provide fun to the players as well as fulfilling a scientific purpose. Second, many annotation tasks can be crowdsourced to collect significant amounts of research data that could not be gathered automatically. These annotation tasks can be distributed among solvers via micro-tasking platforms. In addition, micro-tasking platforms can be used successfully with crowdsourced serious games or with any general crowdsourced application that requires the development of a separate application. Finally, some available platforms support the organization of crowdsourced challenges. These include general-purpose platforms such as DREAM Challenges, which facilitate the organization of any type of challenge, or dedicated platforms such as Trochilus and Optil.io, which use an online judge architecture to allow the crowdsourcing-based development of algorithms.

Based on this wide range of applications, it is possible to define several guidelines to facilitate the design of successful platforms that utilize crowdsourcing. These guidelines are presented in the following where we focus on the task of modeling biological systems. However, most of them should also apply to other areas of science.

Guideline #1: Communicating the goal Crowdsourced users are generally unskilled people who have no detailed domain knowledge, but they usually want to know the scientific objective of the platform. Informing users about this objective using simple and understandable language can convince them to spend more time

solving tasks because they will identify with the mission of the platform. Having a sense of mission is especially important in the case of biomedical research because the users can readily appreciate the connection between solving the problem and the potential benefits of medical discoveries, thereby motivating them to work more intensively on the crowdsourcing platform. This guideline is implemented in all successful crowdsourcing applications, including *Throw the Hamster* games, FoldIt, EyeWire, and DREAM Challenges. Information about the goal is usually provided in a brief tutorial, which the user must read before they start using the application to ensure that it is read. Moreover, more detailed descriptions (including some scientific facts) are usually provided in the documentation that supplements the application.

Guideline #2: Hiding domain knowledge To encourage many users to solve crowdsourced problems, the tasks should be as easy to solve as possible and the domain knowledge should be hidden from the user. This guideline is especially important when modeling biological systems because most people have only elementary and insufficient biological knowledge. After hiding the domain knowledge, it is possible to recruit many solvers from an unskilled crowd. This guideline was one of the central assumptions for the *Throw the Hamster* games. In these games, the construction of mathematical formulae is performed by simply dragging blocks that represent various operations. The players assessed the solutions very successfully. In addition, EyeWire smartly hides medical knowledge about the shapes of neurons from the user by presenting tasks in the form of a coloring page.

Clearly, this guideline has to be interpreted in different ways when designing platforms that publish challenges solved by domain experts, such as Trochilus or DREAM Challenges. In this case, all of the domain knowledge must be described precisely because it can help the participants to develop better solutions. However, the platform should simplify the solving of tasks as much as possible, such as by sharing visualizers or modules for implementing the standard code that must be used by all participants.

Guideline #3: Ease of understanding After hiding the domain knowledge from users, the mechanics of the application may become too difficult for the user to understand. In this case, a user may conclude that the rules are magical and give up solving the tasks. This behavior was a major problem in the first version of the *Throw the Hamster* game because the users could not understand how modifying the tree representing the formula influenced the flight path. We solved this problem by providing general descriptions of how each element worked and by adding a preview of the plot of the constructed function. According to the survey conducted at the end of the experiment, these changes significantly helped users to understand how the game worked. In general, it is better to reveal elementary biological or mathematical knowledge related to the modeled system rather than hide everything possible.

Guideline #4: User friendliness User friendliness is a feature that should characterize every computer application. However, in the case of crowdsourcing, it is especially important because crowdsourced users generally receive no direct benefits from solving the tasks. The user will rapidly stop using the application if it is difficult

to use or it does not look attractive. In biomedical applications, a well presented visualization could also have additional advantages. In particular, it is generally difficult for the user to imagine certain biological concepts such as cells or genetic sequences because they are usually unaware of their appearance. Therefore, a suitably designed visual layer can facilitate access to the application.

This guideline is also partially correct even for micro-tasking websites that pay users for solving tasks. A good example is Amazon's Mechanical Turk, which has a non-intuitive registration process that includes user validation that is difficult to pass. This architecture discourages many users from accessing this platform.

Guideline #5: Automation It is essential to automate the crowdsourced problem solving steps that can be implemented using currently available algorithms and methods. This saves the valuable time spent by the user solving tasks that cannot be solved automatically and that actually require their attention. A good example is the EyeWire game, which uses a flood fill algorithm to mark the parts of neurons that actually belong to the cell according to the algorithm. In addition, FoldIt implements automated protein shape improvement and the *Race the Hamster* game implements automated parameter value tuning. These enhancements significantly improve the efficiency of the work conducted by crowdsourced users and are generally easy to implement, especially in the mathematical components of the modeling applications.

Guideline #6: Decomposition The problem should be decomposed into as many simple tasks as possible, thereby allowing users to be rewarded more often because it is easier to accomplish the task. However, this is not always possible. For example, in the FoldIt game, it is not possible to split an amino acid sequence into subsequences because the tertiary structure of the protein always depends on the whole sequence. Moreover, in the EyeWire games, developers can divide the analyzed parts of the brain into small cubes for separate handling, thereby allowing the generation of rank lists for players based on the number of cubes solved.

Guideline #7: Validation In many cases, it is relatively easy to obtain a biological model from the crowd that is very complex and it may be over-fitted to the test data. Thus, a validation phase should always be conducted to test the collected solutions against validation data. However, in many cases, it may be challenging to define a validation procedure, particularly when the modeling process is divided into many micro-tasks and the validation phase cannot be conducted before all of the micro-tasks are solved. Therefore, mechanisms should be added to ensure that the collected partial solutions are sufficiently simple to prevent over-fitting. This guideline can be especially difficult to implement when using crowdsourcing for modeling biological systems because it is usually difficult to collect sufficient test data. However, this is not sufficient reason to omit a validation phase.

Guideline #8: Verification Crowdsourcing is prone to errors. When outsourcing work to unskilled workers, they can submit an incorrect solution. This is especially true for biomedical applications characterized by many domain-specific constraints.

Thus, a verification procedure must be defined to address this problem. In the case of the *Throw the Hamster* game, the correctness of the models were verified automatically, and if the player did not include the required variables or constants in the formula they were automatically added. In the case of the FoldIt game, manual verification was applied where domain experts verified each promising solution. Finally, micro-tasking websites usually verify solutions by submitting the same task several times to different users. However, regardless of the verification procedure employed, at least one should be implemented.

Guideline #9: Wisdom of the crowd Galton [23] and many of his followers suggested that combining the contributions of many participants can significantly increase the quality of the final solution. This hypothesis was clearly supported by the *Throw the Hamster* game, where the possibility of employing solutions submitted by other users helped to achieve much better results [68]. Unfortunately, most of the implementations of crowdsourcing still fail to exploit this beneficial effect. Accordingly, much more attention should be paid to this aspect of crowdsourcing.

Guideline #10: Rewarding users A good motivation technique is the key to recruiting people to use crowdsourcing applications. Paying money is certainly the easiest method and the *Race the Hamster* game demonstrated that it is feasible to integrate a crowdsourcing application with a micro-tasking platform to support the provision of payments for solving tasks. However, it is also possible to use other cheaper motivation methods, such as gamification elements or forcing people to solve crowdsourcing tasks to receive access to some resources. More information about motivating users to participate in crowdsourcing actions was given by Doan and Bynghall [17].

The best evidence that the guidelines described above are crucial for designing successful crowdsourcing platform is our experience with games dedicated to modeling dynamic systems. In particular, we found that certain elements were lacking in the first version of *Throw the Hamster* game. However, implementing them in the second version of the game, and in the *Race the Hamster* game, significantly increased the quality of the solutions submitted by the players. First, the game became much more interesting, which increased the average number of games played by each player by over four times. Second, the lack of domain knowledge was less of a concern for the players. Finally, they were able to handle more complex models and generate higher quality solutions.

Acknowledgments

The author was supported by the Polish National Center for Research and Development grant no. LIDER/004/103/L-5/13/NCBR/2014. I would also like to thank the master's students who helped me to implement the ideas presented in this review

as part of their master's thesis: Jaroslaw Wulnikowski and Jakub Krzyskow [72], Filip Fratczak [20], and Pawel Wawrzyniak [71], and the team of bachelor's students comprising Iwo Bladek, Tomasz Chojnacki, Konrad Miazga, and Jakub Szwachla [6].

References

[1] An G., Mi Q., Dutta-Moscato J., and Vodovotz Y. Agent-based models in translational systems biology. *Wiley Interdisciplinary Reviews: Systems Biology and Medicine*, 1(2):159–171, 2009.

[2] Ansari S., Binder J., Boue S., Fabio A. D., Hayes W., Hoeng J., Iskandar A., Kleiman R., Norel R., O'neel B., Peitsch M. C., Poussin C., Pratt D., Rhrissorrakrai K., Schlage W. K., Stolovitzky G., and Talikka M. On crowd-verification of biological networks. *Bioinformatics and Biology Insights*, 7:BBI.S12932, 2013.

[3] Bae J. A., Mu S., Kim J. S., Turner N. L., Tartavull I., Kemnitz N., Jordan C. S., Norton A. D., Silversmith W. M., Prentki R., et al. Digital museum of retinal ganglion cells with dense anatomy and physiology. *bioRxiv*, page 182758, 2018.

[4] Barone J., Bayer C., Copley R., Barlow N., Burns M., Rao S., Seelig G., Popovic Z., Cooper S., and Players N. Nanocrafter: Design and evaluation of a dna nanotechnology game. In *Proceedings of the 10th International Conference on the Foundations of Digital Games*, pages 1–5, 2015.

[5] Barrett T., Wilhite S. E., Ledoux P., Evangelista C., Kim I. F., Tomashevsky M., Marshall K. A., Phillippy K. H., Sherman P. M., Holko M., et al. Ncbi geo: archive for functional genomics data sets—update. *Nucleic acids research*, 41(D1):D991–D995, 2012.

[6] Bladek I., Chojnacki T., Miazga K., and Szwachla J. Platform for evaluation of algorithms estimating real paremeters in biological systems (Platforma do oceny algorytmow wyznaczania paramterow rzeczywistych w systemach biologicznych). Master's thesis, Faculty of Computing, Poznan University of Technology, 2014.

[7] Bonabeau E. Decisions 2.0: The power of collective intelligence. *MIT Sloan management review*, 50(2):45, 2009.

[8] Bruggeman F. J. and Westerhoff H. V. The nature of systems biology. *Trends in Microbiology*, 15(1):45–50, 2007.

[9] Cookson C. Online gaming yields first results for alzheimer's research. *Financial Times*, 2017.

[10] Cooper S., Khatib F., Treuille A., Barbero J., Lee J., Beenen M., Leaver-Fay A., Baker D., Popović Z., et al. Predicting protein structures with a multiplayer online game. *Nature*, 466(7307):756, 2010.

[11] Costello J. C. and Stolovitzky G. Seeking the Wisdom of Crowds Through Challenge-Based Competitions in Biomedical Research. *Clinical Pharmacology & Therapeutics*, 93(5):396–398, feb 2013.

[12] Dahari H., Ribeiro R. M., and Perelson A. S. Triphasic decline of hepatitis C virus RNA during antiviral therapy. *Hepatology*, 46(1):16–21, 2007.

[13] Dawson R. and Bynghall S. *Getting Results From Crowds: The definitive guide to using crowdsourcing to grow your business.* Advanced Human Technologies Incl, 2012.

[14] Deng N., Galliers R., and Joshi K. Crowdworking—a new digital divide? is design and research implications. In *ECIS Conference Proceedings*, 2016.

[15] Derry J. M. J., Mangravite L. M., Suver C., Furia M. D., Henderson D., Schildwachter X., Bot B., Izant J., Sieberts S. K., Kellen M. R., and Friend S. H. Developing predictive molecular maps of human disease through community-based modeling. *Nature Genetics*, 44(2):127–130, jan 2012.

[16] DiStefano III J. *Dynamic systems biology modeling and simulation.* Academic Press, 2015.

[17] Doan A., Ramakrishnan R., and Halevy A. Y. Crowdsourcing systems on the world-wide web. *Commun. ACM*, 54(4):86–96, Apr. 2011.

[18] Eiben C. B., Siegel J. B., Bale J. B., Cooper S., Khatib F., Shen B. W., Stoddard B. L., Popovic Z., and Baker D. Increased Diels-Alderase activity through backbone remodeling guided by foldit players. *Nature biotechnology*, 30(2):190, 2012.

[19] Estellés-Arolas E. and González-Ladrón-De-Guevara F. Towards an integrated crowdsourcing definition. *J. Inf. Sci.*, 38(2):189–200, Apr. 2012.

[20] Fratczak F. Design of the algorithm for estimating values of parameters in biological models (Opracowanie algorytmu wyznaczajacego wartosci parametrow w modelach biologicznych). Master's thesis, Faculty of Computing, Poznan University of Technology, 2016.

[21] Fries P. A mechanism for cognitive dynamics: neuronal communication through neuronal coherence. *Trends in Cognitive Sciences*, 9(10):474 – 480, 2005.

[22] Frings M., Berkels B., Behr M., and Elgeti S. Comparison of optimization algorithms for the slow shot phase in hpdc. *AIP Conference Proceedings*, 1960(1):110005, 2018.

[23] Galton F. Vox populi (the wisdom of crowds). *Nature*, 75(7):450–451, 1907.

[24] Good B. M. and Su A. I. Crowdsourcing for bioinformatics. *Bioinformatics*, 29(16):1925–1933, 2013.

[25] Haklay M. and Weber P. Openstreetmap: User-generated street maps. *Ieee Pervas Comput*, 7(4):12–18, 2008.

[26] Helmstaedter M., Briggman K. L., Turaga S. C., Jain V., Seung H. S., and Denk W. Connectomic reconstruction of the inner plexiform layer in the mouse retina. *Nature*, 500(7461):168, 2013.

[27] Hill S. M., Heiser L. M., Cokelaer T., Unger M., Nesser N. K., Carlin D. E., Zhang Y., Sokolov A., Paull E. O., Wong C. K., et al. Inferring causal molecular networks: empirical assessment through a community-based effort. *Nature methods*, 13(4):310, 2016.

[28] Hirschman L., Fort K., Boué S., Kyrpides N., Islamaj Doğan R., and Cohen K. B. Crowdsourcing and curation: perspectives from biology and natural language processing. *Database*, 2016:baw115, 2016.

[29] Ho C. C. and Ting C.-Y. Measuring crowd sourced analytics: A review. *International Information Institute (Tokyo). Information*, 19(10B):4891, 2016.

[30] Hodgkin A. L. and Huxley A. F. A quantitative description of membrane current and its application to conduction and excitation in nerve. *J Physiol*, 117(4):500–544, Aug 1952.

[31] Horowitz S., Koepnick B., Martin R., Tymieniecki A., Winburn A. A., Cooper S., Flatten J., Rogawski D. S., Koropatkin N. M., Hailu T. T., et al. Determining crystal structures through crowdsourcing and coursework. *Nature communications*, 7:12549, 2016.

[32] Howe J. The rise of crowdsourcing. *Wired magazine*, 14(6):1–4, 2006.

[33] Kawrykow A., Roumanis G., Kam A., Kwak D., Leung C., Wu C., Zarour E., players P., Sarmenta L., Blanchette M., and WaldispÃŒhl J. Phylo: A citizen science approach for improving multiple sequence alignment. *PLOS ONE*, 7(3):1–9, 03 2012.

[34] Kim J. S., Greene M. J., Zlateski A., Lee K., Richardson M., Turaga S. C., Purcaro M., Balkam M., Robinson A., Behabadi B. F., et al. Space–time wiring specificity supports direction selectivity in the retina. *Nature*, 509(7500):331, 2014.

[35] Kitano H. *Foundations of Systems Biology*. The MIT Press, 2001.

[36] Kittur A., Chi E. H., and Suh B. Crowdsourcing user studies with mechanical turk. In *Proceedings of the SIGCHI conference on human factors in computing systems*, pages 453–456. ACM, 2008.

[37] Lee J., Kladwang W., Lee M., Cantu D., Azizyan M., Kim H., Limpaecher A., Gaikwad S., Yoon S., Treuille A., Das R., and . Rna design rules from a massive open laboratory. *Proceedings of the National Academy of Sciences*, 111(6):2122–2127, 2014.

[38] Lee J. and Seo D. Crowdsourcing not all sourced by the crowd: An observation on the behavior of wikipedia participants. *Technovation*, 55:14–21, 2016.

[39] Liang J., Qu B., Suganthan P., and Hernández-Díaz A. G. Problem definitions and evaluation criteria for the CEC 2013 special session on real-parameter optimization. *Computational Intelligence Laboratory, Zhengzhou University, Zhengzhou, China and Nanyang Technological University, Singapore, Technical Report*, 201212:3–18, 2013.

[40] Logas H., Whitehead J., Mateas M., Vallejos R., Scott L., Shapiro D., Murray J., Compton K., Osborn J., Salvatore O., and others. Software Verification Games: Designing Xylem, The Code of Plants. In *Foundations of Digital Games 2014 (FDG2014)*, pages 1–8. Center for Games and Playable Media, 2014.

[41] Luengo-Oroz A. M., Arranz A., and Frean J. Crowdsourcing malaria parasite quantification: An online game for analyzing images of infected thick blood smears. *J Med Internet Res*, 14(6):e167, Nov 2012.

[42] Marbach D., Costello J. C., Küffner R., Vega N. M., Prill R. J., Camacho D. M., Allison K. R., Aderhold A., Bonneau R., Chen Y., et al. Wisdom of crowds for robust gene network inference. *Nature methods*, 9(8):796, 2012.

[43] Mavandadi S., Feng S., Yu F., Dimitrov S., Yu R., and Ozcan A. Biogames: A platform for crowd-sourced biomedical image analysis and telediagnosis. *Games for Health Journal*, 1(5):373–376, 2012. PMID: 23724363.

[44] Mesarovic M. *Systems Theory and Biology*. Springer Verlag, 1968.

[45] Meyer P., Cokelaer T., Chandran D., Kim K. H., Loh P.-R., Tucker G., Lipson M., Berger B., Kreutz C., Raue A., Steiert B., Timmer J., Bilal E., Sauro H. M., Stolovitzky G., and Saez-Rodriguez J. Network topology and parameter estimation: from experimental design methods to gene regulatory network kinetics using a community based approach. *BMC Systems Biology*, 8(1):13, Feb 2014.

[46] Mietchen D., Wodak S., Wasik S., Szostak N., and Dessimoz C. Submit a topic page to plos computational biology and wikipedia. *PLOS Computational Biology*, 14(5):1–4, 05 2018.

[47] Nurse P. Reductionism: The ends of understanding. *Nature*, 387(6634):657, 1997.

[48] Peer E., Samat S., Brandimarte L., and Acquisti A. Beyond the turk: An empirical comparison of alternative platforms for crowdsourcing online behavioral research. *NA - Advances in Consumer Research*, 43:18–22, 2016.

[49] Pico A. R., Kelder T., van Iersel M. P., Hanspers K., Conklin B. R., and Evelo C. Wikipathways: Pathway editing for the people. *PLOS Biology*, 6(7):1–4, 07 2008.

[50] Powell M. J. A view of algorithms for optimization without derivatives. *Mathematics Today-Bulletin of the Institute of Mathematics and its Applications*, 43(5):170–174, 2007.

[51] Prejzendanc T., Wasik S., and Blazewicz J. Computer representations of bioinformatics models. *Current Bioinformatics*, 11(5):551–560, 2016.

[52] Prelec D., Seung H. S., and McCoy J. A solution to the single-question crowd wisdom problem. *Nature*, 541(7638):532, 2017.

[53] Saez-Rodriguez J., Costello J. C., Friend S. H., Kellen M. R., Mangravite L., Meyer P., Norman T., and Stolovitzky G. Crowdsourcing biomedical research: leveraging communities as innovation engines. *Nature Reviews Genetics*, 17(8):470–486, jul 2016.

[54] Schmidt M. and Lipson H. Distilling Free-Form Natural Laws from Experimental Data. *Science*, 324(5923):81–85, apr 2009.

[55] Schrier K. Investigating typologies of games as research environments. In *Proceedings of International Conference on the Foundations of Digital Games*, pages 1–4, 2017.

[56] Szostak N., Synak J., Borowski M., Wasik S., and Blazewicz J. Simulating the origins of life: The dual role of RNA replicases as an obstacle to evolution. *PLOS ONE*, 2017.

[57] Szostak N., Wasik S., and Blazewicz J. Hypercycle. *PLOS Computational Biology*, 12(4):e1004853, apr 2016.

[58] Szuba T. T., Polański P., Schab P., and Wielicki P. On efficiency of collective intelligence phenomena. In *Transactions on computational collective intelligence III*, pages 50–73. Springer, 2011.

[59] Tellioglu U., Xie G. G., Rohrer J. P., and Prince C. Whale of a crowd: Quantifying the effectiveness of crowd-sourced serious games. In *2014 Computer Games: AI, Animation, Mobile, Multimedia, Educational and Serious Games (CGAMES)*, pages 1–7, July 2014.

[60] Tomita, Hashimoto, Takahashi, Shimizu, Matsuzaki, Miyoshi, Saito, Tanida, Yugi, Venter, and Hutchison. E-CELL: Software environment for whole cell simulation. *Genome Inform Ser Workshop Genome Inform*, 8:147–155, 1997.

[61] Tyagi M., Hashimoto K., Shoemaker B. A., Wuchty S., and Panchenko A. R. Large-scale mapping of human protein interactome using structural complexes. *EMBO reports*, 13(3):266–271, 2012.

[62] Van Pelt C. and Sorokin A. Designing a scalable crowdsourcing platform. In *Proceedings of the 2012 ACM SIGMOD International Conference on Management of Data*, SIGMOD '12, pages 765–766, New York, NY, USA, 2012. ACM.

[63] Von Ahn L., Maurer B., McMillen C., Abraham D., and Blum M. recaptcha: Human-based character recognition via web security measures. *Science*, 321(5895):1465–1468, 2008.

[64] Wang Z., Monteiro C. D., Jagodnik K. M., Fernandez N. F., Gundersen G. W., Rouillard A. D., Jenkins S. L., Feldmann A. S., Hu K. S., McDermott M. G., et al. Extraction and analysis of signatures from the gene expression omnibus by the crowd. *Nature communications*, 7:12846, 2016.

[65] Wasik S., Antczak M., Badura J., and Laskowski A. Evaluation as a service architecture and crowdsourced problems solving implemented in optil. io platform. *arXiv preprint arXiv:1807.06002*, 2018.

[66] Wasik S., Antczak M., Badura J., Laskowski A., and Sternal T. Optil.io: Cloud Based Platform For Solving Optimization Problems Using Crowdsourcing Approach. In *Proceedings of the 19th ACM Conference on Computer Supported Cooperative Work and Social Computing Companion*, pages 433–436, San Francisco, CA, USA, 2016. ACM, ACM Digital Library.

[67] Wasik S., Antczak M., Badura J., Laskowski A., and Sternal T. A survey on online judge systems and their applications. *ACM Comput. Surv.*, 51(1):3:1–3:34, Jan. 2018.

[68] Wasik S., Fratczak F., Krzyskow J., and Wulnikowski J. Inferring Mathematical Equations Using Crowdsourcing. *PLOS ONE*, 10(12):e0145557, dec 2015.

[69] Wasik S., Jackowiak P., Figlerowicz M., and Blazewicz J. Multi-agent model of hepatitis C virus infection. *Artificial Intelligence in Medicine*, 60(2):123–131, feb 2014.

[70] Wasik S., Prejzendanc T., and Blazewicz J. ModeLang - a new approach for experts-friendly viral infections modeling. *Computational and Mathematical Methods in Medicine*, 2013:8, 2013.

[71] Wawrzyniak P. Methods of integrating micro-tasking websites with crowdsourced serious games (Metody integracji serwisow mikro-zadaniowych z grami komputerowymi adresowanymi do tlumow). Master's thesis, Faculty of Computing, Poznan University of Technology, 2018.

[72] Wulnikowski J. and Krzyskow J. Design and implementation of the game utilizing crowdsourcing to model dynamic systems (Opracowanie gry wykorzystujacej crowdsourcing do modelowania systemow dynamicznych). Master's thesis, Faculty of Computing, Poznan University of Technology, 2016.

[73] Xintong G., Hongzhi W., Song Y., and Hong G. Brief survey of crowdsourcing for data mining. *Expert Systems with Applications*, 41(17):7987–7994, 2014.

[74] Zhan C. and Yeung L. F. Parameter estimation in systems biology models using spline approximation. *BMC Systems Biology*, 5(1):14, Jan 2011.

Received 30.07.2018, Accepted 07.09.2018