GPU-Based Bat Algorithm for Discovering Cultural Coalitions

Amine Kechid(✉) and Habiba Drias

Laboratory for Research in Artificial Intelligence, Computer Science Department, USTHB, BP 32 El Alia Bab Ezzouar, 16111 Algiers, Algeria
{akechid,hdrias}@usthb.dz

Abstract. Nowadays, artificial intelligence makes a great success in our modern social life. The human should be prepared to be able to live with social robots that can provide him comfort and help in solving complex processes. Extending the use of robot's technology is certainly desirable, but preventing certain catastrophes from the misdeeds of artificial intelligence is crucial. One of these troubles could be for instance the creation of robots' coalitions to impose pernicious decisions. As a contribution to cope with such issue, we propose a parallel approach for the detection of cultural coalitions based on Bat Algorithm. This GPU-based bat algorithm approach can treat very large datasets due to the possibility of launching several artificial bats simultaneously, which contribute to reducing the runtime without affecting the performance. To proof the effectiveness of the parallel detection coalition method, we conducted several experiments on datasets of different sizes. These datasets represent the result of cultural artificial agents playing the colored trails (CT) game. For the creation of agents' profiles, we use real cultural datasets generated based on the WV survey. The experimental analysis demonstrates that the use of the proposed method will considerably reduce the runtime.

Keywords: Culture · Coalitions · Social agents · Detecting coalitions

1 Introduction

Few years ago, robots were just a grain of imagination, but with the great advancement in the technological domain, this imagination becomes a reality. Robots were created to facilitate difficult tasks such that we find in industry, military, and health domain. Due to their great success, researchers seek to improve their physical appearance until getting robots made from a special batter that are difficult to distinguish from humans. As a result, robots crawl into human social life. For the best interaction between robots and humans, researchers aim to increase the social intelligence of robots by cloning the human cultural intelligence and transplanting it into the artificial intelligence of the robot.

F. Wotawa et al. (Eds.): IEA/AIE 2019, LNAI 11606, pp. 470–482, 2019.
https://doi.org/10.1007/978-3-030-22999-3_41

As each technology, robot engineering has advantages and drawbacks. While its positives and favor over humanity are great, its drawbacks are also great and could lead to a major scientific disaster. One of these weaknesses could be the creation of coalitions of robots to impose pernicious decisions. The issue is that a subset of users prefer to cooperate within the group and avoid cooperation outside the group [2], such that the elements of the same coalition share the same interests and objectives [19]. As knowing the causes of a problem is the most effective solution to avoid it, detecting coalition becomes nowadays an essential solution to avoid an inevitable Disaster.

The issue of finding coalitions is a complex problem and classified in the NP-hard category [22]. It means that this problem lacks from a resolution algorithm of polynomial complexity, which makes finding solutions needs exponential running time. Techniques of artificial intelligence have proved their effectiveness in solving such problems with approximate methods. Unlike the exact methods that find all the coalitions, intelligent tools find the most efficient ones in a significantly reduced amount of time. Swarm algorithms are one of these methods that are nature-inspired, mimicking some successful characteristics of animal swarms. It is a promising field based on the simulation of the collective intelligence in social insects and some animals such as fishes and birds. The most known algorithms in this category are Ant Colony Optimization (ACO) [4], Particle Swarm Optimization (PSO) [11], Firefly Algorithm (FA) [24], and Bat Algorithm (BA), a meta-heuristic developed in 2010 [25]. This approach simulates the movements of bats when searching for preys. It manages a balance between a local search and PSO optimization, and is appreciated for its good convergence to the optimal solutions.

Although these methods are basically efficient, however with the use of very large datasets the time complexity issue reappears. One way to tackle this hard question is to parallelized the independent parts. Since 2007 and due to the advancement in the graphics hardware, it was possible to exploit this idea with the emergence of the Graphics Processor Units (GPU). This type of processor differs with the old processor in fundamental design philosophy, where multiple threads cooperate to achieve the goal. For example, in the case of matrix multiplications, instead of filing the matrix element by element, all the elements will be filed in a parallel manner. Each thread is associated with one element, where it calculates the sum of the product of all the elements from the row i with all the elements from the column j. This kind of programming can achieve more than 100 times speedup over sequential execution.

In this paper, we propose a parallel approach for the detection of cultural coalitions based on the Bat Algorithm. It aims to perform more efficiently than the existing state-of-the-art algorithms. This approach needs a GPU architecture and some synchronizations with the CPU, following the master/workers model. Technically, the designed method consists in running the master on the CPU, which generates a population of n bats, where each one represents a potential solution. Then, the master launches n threads offloaded on the GPU, where each one performs the tasks of one bat to improve its solution by sharing its

best solution with its congeners. The master attempts to receive the result of each bat to rank the bats on the best solutions in the next iteration.

The remainder of the paper is structured as follows. In the next section, we present the related works. Then, in Sect. 3, we present our first contribution on the perception of culture and its integration in the BDI model. In Sect. 4, we present our second contribution, which deals with the adaptation of Bat Algorithm to the coalition detection problem. After that, we present the parallel bat algorithm for the detection of coalitions. Finally, we end this paper with a conclusion after presenting the experimental results.

2 Literature Review

Coalition creation may cause serious problems such as competitive harm, the trust and the reputation abuse. To tackle this issue several approaches were proposed using methods based on Markov chain [3], clustering [23], Rule based, graph based [9], machine learning, convolutional neural networks [1]. In 2007, Metwally and his co-authors proposed a method for detecting coalition in advertising networks named detectives [15]. This method used a sampling approach to detect all pairs of publishers having similarity exceeding a specific threshold. In order to detect coalition members, the authors used these pairs of publishers to construct a similarity graph. The graph is composed of vertices which represent publishers. Two vertices are related with edge if the similarity of visitors between the two vertices is greater than or equal to a specific threshold. After creating the similarity graph, they try to detect the maximal clique. Finding a clique in this graph amounts to finding a set of vertices that are related to each other.

After that, other researchers proposed CATCH Algorithm [14], which can detect coalition groups based on the ratio to gain of cost. A coalition is defined as a group that has its gain per resource (GPR) exceeds a specific threshold and every subgroup has its gain per resource lower than the gain per resource of the group.

After one year, Kerr and Cohen proposed a statistical approach [12], which consists of two steps, clustering and characterizing clusters which contain coalition. In the first step, they use the k-means algorithm [6] to regroup agent interest in several clusters. Agent interest is a vector of n entities which represents the interest of the agent with each other. Coalition members are seen as elements that have a greater interest with coalition members than other members. Therefore, in the second step, the average benefit of each candidate cluster is calculated. After that, for each candidate cluster, they take 100 samples of the same size and calculate the benefit between this cluster and each sample to generate large data. From this data, they use the normal distribution to detect whether the probability of the average benefit calculated above is greater than the threshold.

Another method was also proposed to detect the stock market colluding groups based on spectral clustering algorithm [18]. The authors run several times the k-means algorithm until maximizing a certain modularity function. In each iteration, the number of clusters is incremented. On the other hand, the colluders similarity measure (CSM) [16] uses Colluders Detection Algorithm (CDA) on the list of suspicious nodes to cluster the nodes with similarity more than a specific threshold. Another based clustering method was proposed to detect fraud in internet advertising [21]. It consists of 3 steps: constructing, clustering and filtering. At the first step, they organize the data, then, they transform the coalition detection to a clustering problem and finally, they remove false alarm clusters.

Hybridizing the similarity [15] and GPR [14] methods is the subject of a new approach used for detecting coalition in online advertising [27]. This approach consists of 3 phases: initialization, inductive and finalization. The initialization phase generates the required information in the next step to each metric. In the inductive phase, the method uses the apriori style to generate the candidate coalitions, which are transmitted to each method. The result intersection for each method is used to calculate the candidate coalition for the next iteration. The final coalition is identified in the last step as the intersection of the identified coalition in each method.

In 2017, Zhai and his co-authors proposed an approach to detect potential collusive clique with their activities [26]. This method starts by estimating the probability of being fraud to each potential candidate. Then, if confirmed fraudulent, their potential wealth is calculated.

3 Culture and Its Integration in the BDI Model

In this section, we present the needed background to understand the remainder of the paper. We start by presenting our modeling of culture. Then, we explain how to integrate the concept of culture in the BDI model.

3.1 Our Perception of Culture and Modeling

In previous works [10], we define culture as a multivariate mathematical function, that assigns each tuple $(x_1, x_2, ..., x_n)$ in domain D a class $C = f(x_1, x_2, ..., x_n)$:

$$\begin{array}{rccc} f: & D & \rightarrow & R \\ & (x_1, x_2, ..., x_n) & \mapsto & f(x_1, x_2, ..., x_n) \end{array} \tag{1}$$

The tuple is viewed as a vector of socio-cultural factors which is made up of the following elements presented in Table 1. For each factor, we select one or more attributes from the WV survey. This survey is started in 1981 and accessible from the www.worldvaluessurvey.org website. For each country, a dataset is generated, where each instance represents the answers of one person from this country.

After that, we move to the creation of cultural datasets based on the data generated from the survey. From each of the six following countries: Algeria, Germany, China, Japan, Spain and the United States, we create a cultural dataset, where rows represent the individual answers on the selected attributes. These six datasets are analyzed with the Apriori algorithm to generate the frequent cultural characteristics in each region and extract the cultural association rules.

3.2 Culture Integration in the Belief-Desire-Intention (BDI) Model

After we have introduced the culture modeling, we present the integration of this concept in the BDI model [17] to implement multi-agent systems [20]. Cultural attributes are seen as a knowledge base, which can be modified through the encountered events.

The BDI model is used in the creation of rational agents. It consists of 3 concepts: Beliefs which represent the agent knowledge in his environment, desires are agent motivations that represent states of the world that agent wants to reach, and intentions are states of the world that agent undertakes to realize at a given moment. When an agent detects a new event, he updates his beliefs according to the perceived event and his culture to generate an options' list, which represents the objectives that can be instantiated. From these desires, he selects the best choice which represents his intention.

4 Bat Algorithm for Coalitions' Detection

BA is a generic algorithm that can find solutions for several complex issues [8,13]. In this section, we present our previous work, which adapts the BA algorithm to the problem of finding coalitions. We present the formulation of the coalition problem, the solution representation, and the fitness function for the performance evaluation of the artificial bats that encapsulate solutions.

4.1 Problem Formulation

We modeled the problem of finding coalitions as a simple undirected graph G = (V; E), where each vertex of V represents a specific entity (individual) in the system, and there is an edge between two vertices i and j of V if the similarity between the profile interest of the two nodes is less than a specific threshold.

The profile interest of an entity i is a vector of size k, where each dimension j represents the amount of interest between the entity i and j, and k represents the number of entities in the system.

For this problem, we can find one or more solutions. A solution is a set of entities that each one has an edge with each other in the same coalition.

Table 1. Culture dataset attributes

Contribution name	Attribute	Possible values
Reading	Newspaper use	Weekly, Monthly, Less than monthly, Never, No answer
	The use of magazines	
Tradition	Tradition importance	Very Important, Important, Somewhat Important, A little Important, Not Important, Not at all Important, No answer
Age	Age category	Child, Teen, Adult, Old person
Work	Work importance	Very important, Rather important, Not very important, Not at all important, No answer, Don't know
Educational	Highest educational level	From Incomplete primary school to University - level education, with degree, No answer
The interaction with nearby environment	Friends importance	Very important, Rather important, Not very important, Not at all important, No answer, Don't know
Family	Family importance	Very important, Rather important, Not very important, Not at all important, No answer, Don't know
	Teach independence to children	Very important, not very important
	Teach hard work to children	
	Teach feeling of responsibility to children	
	Teach imagination to children	
	Teach tolerance and respect for other people to children	
	Teach thrift to children	
	Teach determination and perseverance to children	
	Teach religious faith to children	
	Teach unselfishness to children	
	Teach obedience to children	
	Teach self-expression to children	
Individual	think up new ideas	Very Important, Important, Somewhat Important, A little Important, Not Important, Not at all Important, No answer
	be rich	
	Living in secure surroundings	
	Luxury and comfort	
	do something for the good of society	
	help people living nearby	
	Being very successful	
	Adventure and taking risks	
	behave properly	
	Looking after the environment	
Universal	Internet use	Daily, Weekly, Monthly, Less than monthly, Never, No answer
	TV use	

4.2 Solution Representation

In this problem, we aim to find the sets of agents that participate in coalitions. As with the BA algorithm each bat encapsulates a single solution, the most appropriate data structure that represents each bat (solution) is a vector. The elements of this vector can take values from 0 to the number of entities in the system. If the value of the element i is equal to 0, it means that this element is not used. Otherwise, it shows the number of the entity that participates in the coalition. For example, if there are 10 entities in the system, the coalition that contains the entities 1, 9, 10, is represented by: 1 9 10 0 0 0 0 0 0 0

4.3 Fitness Function

As BA can generate a lot of solutions, it is necessary to have an effective fitness function to evaluate the quality of the solution and to guide the future solutions. For this purpose, we propose to use the following function.

$$f(x) = \begin{cases} \text{the size of coalition,} & \text{if all the elements are connected.} \\ 0 & \text{Otherwise} \end{cases}$$

4.4 New Solution Generation

Concerning the generation of the new position, we use Algorithm 1 and Eqs. 2 and 3 [7]. This algorithm aims to update some elements from the solution indicated by the actual frequency. This modification starts at a specific element indicated by the actual velocity. So, for each element, it compares the loudness and a random value. If this value is greater, we increment the value of the bit v_i in the solution and save its modulo (k+1). Otherwise, we decrement the value of the bit v_i and save its modulo. To avoid the redundancy of values in the same solution, we update the content of bit v_i to the calculated value if it does not exist in the generated solution. After that, it remains just to increment the value of v_i for passing to the next iteration. This process is repeated until v_i achieve the actual frequency.

5 Parallel Bat Algorithm for the Detection of Coalitions

As described in the previous section, BA is a very effective algorithm and can find solutions to several problems. However, when dealing with very large datasets, we are obliged to increase the number of bats to obtain the desired performance, which increase considerably the runtime of the algorithm. One way to tackle this issue is to launch all the bats in parallel. It means, that all the bats start at the same time. Exploiting this idea needs a new type of processor named Graphics Processor Units (GPU).

In this section, we propose the parallel coalition detection method based on a master/workers paradigm. Whereas the master is executed on the CPU, the

Algorithm 1. Generate new solution

Require: Coalition x_i^{t-1}, Velocity v_i, Loudness A_i ?
Ensure: New coalition x_i^t,
Begin
while $v_i \prec f_i^t$ **do**
if $rand > A_i^t$ **then**
1- new_entity=((entity at v_i) +1) mod (k+1)
else
2- new_entity=((entity at v_i) -1) mod (k+1)
end if
if new_entity does not exist in x_i^t **then**
3- entity at v_i =new_entity
end if
4- increment v_i
end while

workers are offloaded to GPU. Unlike BA, all the bats perform a local search simultaneously. The master starts by initializing randomly one solution to each bat, and initialize the velocity v_i^0 and the frequency f_i^0. After initializing the value of pulse rate r_i^0 and loudness A_i^0, the population is evaluated to extract the best solution x*.

After that, since the maximum number of iterations is not reached again, it copies the input data to the device (GPU) to launch the necessaries number of threads. This process is explained in Algorithm 2. Afterwards, each thread calculates its real index in the set of data by contribution to its index in the block, the index of the block and the number of threads in the block. Each launched thread performs the process of one bat that is, each bat performs a virtual movement using Algorithm 1 to generate a new solution by adjusting frequency f_i, velocity v_i as shown in Eqs. 2 and 3. After evaluating the new solutions, each bat in the population generates a new solution through a random walk using Eq. 4.

$$f_i^t = 1 + (f_{max}) * \beta \tag{2}$$

$$v_i^t = f_{max} - f_i^t - v_i^{t-1} \tag{3}$$

$$x_{new} = x^{'} + \epsilon * A \tag{4}$$

– Where,
- $x*$ is the current best solution.
- β is a random vector in the range [0,1].
- f_{min} and f_{max} are respectively the specified lower and upper bounds for the frequency parameter f.
- x' is the best solution at the actual iteration.
- ϵ is a random value between 0 and 1.
- A is the average loudness of all the bats at the actual iteration.

Finally, the pulse emission rate r_i and the loudness A_i are updated using Eqs. 5 and 6. This principle is explained in Algorithm 3. After every bat ends its tasks and the master receives the solutions, it sorts the solutions according to its quality and then it ranks the bats of the best solutions

$$A_i^t = \alpha * A_i^{t-1} \tag{5}$$

$$r_i^t = r_i^0(1 - e^{\gamma t}) \tag{6}$$

– Where,
  - α and γ are a constants

Algorithm 2. Pseudo code of the parallel Bat algorithm (master)

Require: The graph G.
Ensure: The sets of coalitions.
Begin
1- generate at random a population of n bats (n solutions);
for each bat i **do**
2- define its loudness A_i, its pulse frequency f_i and its velocity v_i;
3- set its pulse rate to r_i;
4- select the best solution x*;
end for
while (Max-Iter not reached) **do**
5- cudaMemcpy(Population, cudaMemcpyHostToDevice)
6- cudaMemcpy(loudness A, cudaMemcpyHostToDevice)
7- cudaMemcpy(frequency f, cudaMemcpyHostToDevice)
8- cudaMemcpy(velocity v, cudaMemcpyHostToDevice)
9- cudaMemcpy(pulse rate r, cudaMemcpyHostToDevice)
10- Launch n threads
11- cudaMemcpy(Population, cudaMemcpyDeviceToHost)
12- cudaMemcpy(loudness A, cudaMemcpyHostToDevice)
13- cudaMemcpy(frequency f, cudaMemcpyHostToDevice)
14- cudaMemcpy(velocity v, cudaMemcpyHostToDevice)
15- cudaMemcpy(pulse rate r, cudaMemcpyHostToDevice)
16- Rank the bats and find the current best solution x*;
end while

6 Experiments

To appreciate the performance of the proposed method, we conducted several experiments on datasets of different sizes. These datasets are generated in previous work, where we developed a simulation environment that contains intelligent cultural agents playing the colored trails game (CT) [5]. These datasets consist of several instances, where each one represents the benefit between each agent

Algorithm 3. Pseudo code of the parallel Bat algorithm (Kernel))

Require: Population, loudness A, frequency f, velocity v, pulse rate r.
Ensure: Population, loudness A, frequency f, velocity v, pulse rate r.
1 i ←blockIdx.x ∗ blockDim.x + threadIdx.x
2- compute a new solution (f_i, v_i, x_i) using Algorithm 1 and equations 2 and 3.
if $rand \succ r_i$ **then**
3- select a solution x' among the best solutions;
4- improve the solution using equation 4;
end if
5- generate at random a new solution (f_i, v_i, x_i);
if rand $\prec$ f(x*) **then**
6- accept the new solution;
7- increase r_i and reduce A_i using formulas 5 and 6
end if

and each other in the system. The agents' profiles are collected from the culture dataset based on the WV survey www.worldvaluessurvey.org. We implemented the proposed method with the C-CUDA 4.0 language using a CPU host coupled with a GPU device. The CPU is a quad-core Intel.

In these experiments, we used four datasets. The size of the datasets varies from the 1000 to 4000 instances in increment of 1000. For each dataset, we varied the size of the population from 200 to 500 bats. Figure 1 shows the runtime of the parallel and the sequential method on the first dataset. It is easy to notice that the curve of the proposed method falls under that of the bat algorithm based method, which means that the runtime of the GPU-based bat algorithm method is much reduced by contribution to the bat algorithm based method.

Figure 2 shows the same information as the previous figure, but on the second dataset. We can see that even if we increase the size of the dataset, the parallel method is faster than the sequential one.

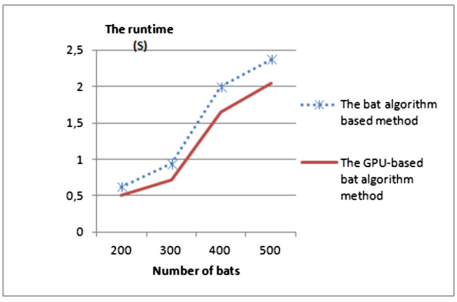

Fig. 1. The runtime of the two methods on the first dataset

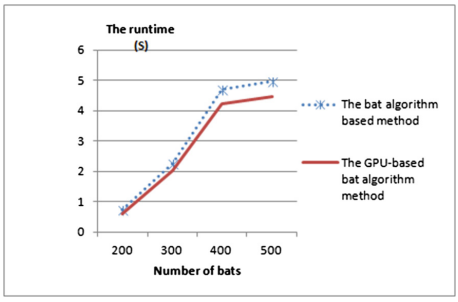

Fig. 2. The runtime of the two methods on the second dataset

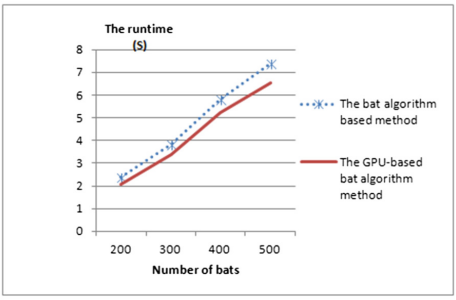

Fig. 3. The runtime of the two methods on the third dataset

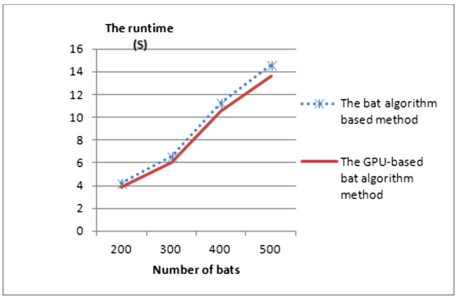

Fig. 4. The runtime of the two methods on the fourth dataset

Figures 3 and 4 show the runtime of the two methods on the third and the fourth dataset, respectively. From these figures, we can confirm the conclusion drawn from the previous experiments, which is the parallel method is faster than the sequential one. So, all the realized experiments share the same conclusion, which is the use of the proposed method reduces the runtime.

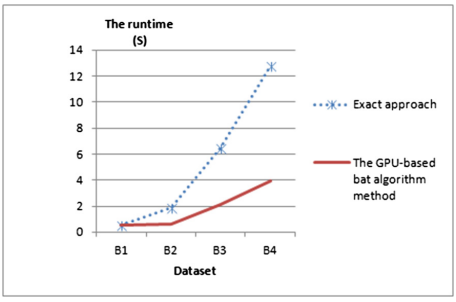

Fig. 5. Comparing the runtime of the proposed method with an exact approach

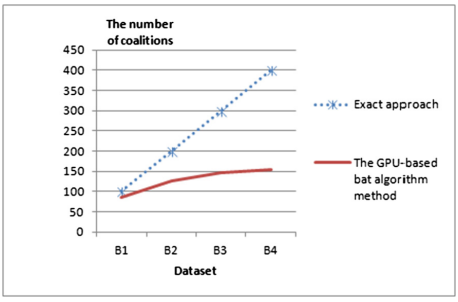

Fig. 6. Comparing the performance of the proposed method with an exact approach

After comparing the parallel and the sequential methods, we pass to compare the proposed method with an exact approach named Similarity-based method [15,27]. Figures 6 and 5 show respectively the performance and the runtime for the two methods on the four datasets. From Fig. 5, we see that the curve of the proposed method is situated bottom the curve of the exact approach, which means that the runtime of the proposed method is well reduced.

On the other hand, from Fig. 6, we see that the curve of the proposed method is situated bottom the curve of the exact approach. It is clear that the performance of the exact approach is higher than the performance of the proposed method, but the performance of the proposed method remains also good.

7 Conclusion

In this paper, we propose a parallel detection coalition method based on Bat Algorithm. This approach allows reducing the runtime while obtaining good results. Unlike the state-of-art methods, our proposal finds the most effective solutions in a significantly reduced amount of time.

When dealing with very large datasets, traditional techniques need to increase the population size, otherwise, we lose in the performance. However, increasing the population's size leads to an increase in the running time. The proposed method allows launching simultaneously several bats on the GPU, which gives good results without affecting the runtime.

To validate the detection coalition method, we implemented our algorithms with C-CUDA 4.0 language on a CPU coupled with a GPU architecture. The obtained result from all the experiments shows the importance of the method.

References

1. Abdallah, A., Maarof, M.A., Zainal, A.: Fraud detection system: a survey. J. Netw. Comput. Appl. **68**, 90–113 (2016)
2. Belmonte, M.V., Conejo, R., Pérez-de-la-Cruz, J.L., Triguero, F.: A stable and feasible payoff division for coalition formation in a class of task oriented domains. In: Meyer, J.-J.C., Tambe, M. (eds.) ATAL 2001. LNCS (LNAI), vol. 2333, pp. 324–334. Springer, Heidelberg (2002). https://doi.org/10.1007/3-540-45448-9_24
3. Davis, B., Conwell, W.: Methods and systems to help detect identity fraud, 12 July 2007, uS Patent App. 11/613,891
4. Dorigo, M., Di Caro, G.: Ant colony optimization: a new meta-heuristic. In: Proceedings of the 1999 Congress on Evolutionary Computation-CEC99 (Cat. No. 99TH8406), vol. 2, pp. 1470–1477. IEEE (1999)
5. Haim, G., An, B., Kraus, S., et al.: Human-computer negotiation in a three player market setting. Artif. Intell. **246**, 34–52 (2017)
6. Han, J., Pei, J., Kamber, M.: Data Mining: Concepts and Techniques. Elsevier, Amsterdam (2011)
7. Heraguemi, K.E., Kamel, N., Drias, H.: Association rule mining based on bat algorithm. In: Pan, L., Păun, G., Pérez-Jiménez, M.J., Song, T. (eds.) BIC-TA 2014. CCIS, vol. 472, pp. 182–186. Springer, Heidelberg (2014). https://doi.org/10.1007/978-3-662-45049-9_29
8. Heraguemi, K.E., Kamel, N., Drias, H.: Multi-swarm bat algorithm for association rule mining using multiple cooperative strategies. Appl. Intell. **45**(4), 1021–1033 (2016)
9. Jiang, M., Cui, P., Beutel, A., Faloutsos, C., Yang, S.: CatchSync: catching synchronized behavior in large directed graphs. In: Proceedings of the 20th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 941–950. ACM (2014)
10. Kechid, A., Drias, H.: Association rules mining for culture modeling. In: Rocha, Á., Adeli, H., Reis, L.P., Costanzo, S. (eds.) WorldCIST'18 2018. AISC, vol. 746, pp. 378–387. Springer, Cham (2018). https://doi.org/10.1007/978-3-319-77712-2_36
11. Kennedy, J.: Particle swarm optimization. In: Sammut, C., Webb, G.I. (eds.) Encyclopedia of Machine Learning, pp. 760–766. Springer, Heidelberg (2011). https://doi.org/10.1007/978-0-387-30164-8_630

12. Kerr, R., Cohen, R.: Detecting and identifying coalitions. In: Proceedings of the 11th International Conference on Autonomous Agents and Multiagent Systems, vol. 3, pp. 1363–1364. International Foundation for Autonomous Agents and Multiagent Systems (2012)
13. Khennak, I., Drias, H.: Bat-inspired algorithm based query expansion for medical web information retrieval. J. Med. Syst. **41**(2), 34 (2017)
14. Kim, C., Miao, H., Shim, K.: Catch: a detecting algorithm for coalition attacks of hit inflation in internet advertising. Inf. Syst. **36**(8), 1105–1123 (2011)
15. Metwally, A., Agrawal, D., El Abbadi, A.: Detectives: detecting coalition hit inflation attacks in advertising networks streams. In: Proceedings of the 16th International Conference on World Wide Web, pp. 241–250. ACM (2007)
16. Niknafs, M., Dorri Nogoorani, S., Jalili, R.: A collusion mitigation scheme for reputation systems. ISC Int. J. Inf. Secur. **7**(2), 151–166 (2015)
17. Rao, A.S., Georgeff, M.P., et al.: BDI agents: from theory to practice. In: ICMAS, vol. 95, pp. 312–319 (1995)
18. Sarswat, S., Abraham, K.M., Ghosh, S.K.: Identifying collusion groups using spectral clustering. arXiv preprint arXiv:1509.06457 (2015)
19. Shehory, O., Kraus, S.: Feasible formation of coalitions among autonomous agents in nonsuperadditive environments. Comput. Intell. **15**(3), 218–251 (1999)
20. Shoham, Y., Leyton-Brown, K.: Multiagent Systems: Algorithmic, Game-Theoretic, and Logical Foundations. Cambridge University Press, Cambridge (2008)
21. Tian, T., Zhu, J., Xia, F., Zhuang, X., Zhang, T.: Crowd fraud detection in internet advertising. In: Proceedings of the 24th International Conference on World Wide Web, pp. 1100–1110. International World Wide Web Conferences Steering Committee (2015)
22. Tomita, E., Tanaka, A., Takahashi, H.: The worst-case time complexity for generating all maximal cliques and computational experiments. Theoret. Comput. Sci. **363**(1), 28–42 (2006)
23. Wang, G., Zhang, X., Tang, S., Zheng, H., Zhao, B.Y.: Unsupervised clickstream clustering for user behavior analysis. In: Proceedings of the 2016 CHI Conference on Human Factors in Computing Systems, pp. 225–236. ACM (2016)
24. Yang, X.S.: Nature-Inspired Metaheuristic Algorithms. Luniver Press (2010)
25. Yang, X.S.: A new metaheuristic bat-inspired algorithm. In: NICSO 2010, vol. 284, pp. 65–74. Springer, Heidelberg (2010)
26. Zhai, J., Cao, Y., Yao, Y., Ding, X., Li, Y.: Coarse and fine identification of collusive clique in financial market. Expert Syst. Appl. **69**, 225–238 (2017)
27. Zhang, Q., Feng, W.: Detecting coalition attacks in online advertising: a hybrid data mining approach. Big Data Inf. Anal. **1**(2/3), 227–245 (2016)