Resource Discovery in Non-Structured Peer to Peer Grid Systems Using the Shuffled Frog Leaping Algorithm

A.Ahmadian[1], M.Zavvar[1], A.Saeedi[2], F.Ramezani[1]
[1]*Department of Computer Engineering, Sari Branch, Islamic Azad University, Sari, Iran.*
[2]*Department of Computer Engineering, Mashhad Branch, Islamic Azad University, Mashhad, Iran.*
Zavvar.developer@gmail.com

***Abstract*—In Peer to Peer (P2P) grid systems, users can utilize the resources of other machines for their tasks without involving themselves in the detailed aspects of addressing. One of the greatest challenges for these systems is finding the resource that matches the user's request to minimize query traffic in the network. Thus, inspired by the Shuffled Frog Leaping Algorithm (SFLA), this article presents a new method for resource discovery in grid systems. This algorithm finds the resource that matches the user's request via sending requests to the most suitable neighbors, thus preventing the flooding of requests and reducing traffic. The evaluation and comparison of the proposed method with the Genetic Algorithm (GA) and Differential Evolution Algorithm (DEA) indicate that it yields higher performance considering the speed and number of sent queries in the network.**

***Index Terms*— Differential Evolution Algorithm; Genetic Algorithmm; Grid Systems; Peer to Peer Systems; Resource Discovery; Shuffled Frog Leaping Algorithm.**

I. INTRODUCTION

The term "grid" was first coined in the mid-1990s for the act of accessing a large collection of heterogeneous resources as a united, integrated resource for solving the issues of large-scale calculations and concentrated data in advanced sciences and engineering [1]. The grid system is a series of distributed hardware and software resources that forms a calculation model and a sharing environment. These resources are remotely accessible for users and the execution of processes is distributed among computers, which enhance processing power [2-5].

In the past, there was a tendency for focusing on two sharing environments that are the grid systems and Peer to Peer (P2P) systems. These two systems have evolved in different ways. Grid systems have been gradually developed for complex systems, connecting a small number of peer sites that are involved in complex scientific projects. On the other hand, P2P connections have been quickly developed for simple but popular services such as file sharing, emphasizing on scalability and user support through serial of communications and resource sharing [6].

One of the main challenges in grid systems is resource discovery, in which suitable resources are found according to the requested task. This problem can also be defined as the searching and locating of candidate resources for performing a task within a reasonable time. Consequently, resource allocation in grid calculation systems is a complex process as they involve dynamic, multi-attribute queries and sharing to meet the demands of users and resource owners [7]. The traditional methods of resource discovery in a grid environment are mainly based on concentrated or hierarchical models. However, due to factors such as breaking point, bottlenecks in a very dynamic environment, system scalability and geographical distribution all over the internet, these methods face limitations in their implementation. Thus, it is important to decentralize the attributes to prevent such problems [8].

Due to the importance of resource discovery in grid systems, a query transportation strategy for the P2P architecture is proposed: Each peer acts independently, and in case it has the required resources, it sends a response message to the requester. Otherwise, it relays the request to the neighboring peer. This paper aims to illustrate the possibility of using Shuffled Frog Leaping Algorithm (SFLA) to obtain a solution for resource discovery in non-structural P2P grid systems. Thus, this paper first reviews the related works in literature and the SFLA, and then proposes a method for resource discovery in non-structures P2P grid systems. In the next section, the proposed algorithm is evaluated and the results are compared to the methods based on Genetic Algorithm (GA) and Differential Evolution Algorithm (DEA). Finally, the directions for future research are presented.

II. RELATED WORKS

Reliability, scalability, and accessibility of a large number of resources in the P2P model have led to the use of its concepts and techniques for implementing non-hierarchical and non-concentrated grid systems. Here, there are many research projects studying P2P techniques for grid systems that are able to share resources distributed by various organizations [9]. In non-structured systems, each peer is randomly connected to a constant number of peers, and there is no information about the location of resources [10].

The learning automata resource discovery (LARD) is one of the non-structured P2P mechanisms developed for the discovery of non-concentrated resources in large-scale networks. This algorithm was proposed to address the issues of former methods. The learning automata are used for finding the shortest route [11].

In [8], researchers proposed a resource discovery method based on GA for finding the required resources in P2P networks. This method prevents the flooding of queries that increases traffic, while decreases the number of queries by

sending them to the best neighboring peers. However, this method is applicable to inconsistent environments where only single attribute queries can be defined.

In another research, Yow et al. proposed a web service based on publication directory service that provides service for clients and resources using web and XML message format. This method provides high-level services and was designed for activating the idea of the economic grid [12].

In [13], the authors proposed a distributed method for addressing the resource discovery issue in a self-structured grid. They applied an ant colony algorithm for the formation and the support of all nodes in a P2P grid network with minimal link count and limited network size.

In [9], the ant colony algorithm was used for a large scale resource discovery in P2P grid systems. In this mechanism, each ant represents a query message. At the beginning, ants randomly move from nest to nest to locate the resources. If they find the required resources, they return to their original nest from the same route and update routing information in their memory.

Another group of researchers developed a model for a P2P information grid service [14]. This model consists of two peers: the super peer and the regular peer. Each super grid connects to a number of local regular peers. Super peers are connected via an overlapping P2P network. The regular peer sends its requests to the local super peer.

Papadakis et al. [15] implemented a P2P grid resource discovery system which support both static and dynamic information retrieval modes, and the "push and pull" model. For static information, the search was performed similar to the structured method, while for dynamic information, the information search was similar to a functional non-structured method in accordance with the DHT structure.

Ebadi & Khanali proposed a new distributed, hierarchical mechanism for service discovery in a grid environment in order to improve fault tolerance and service discovery speed [16]. In this mechanism, the architecture of resource discovery consists of five layers: client and service, institution, organization, domain, and root layers. In this architecture, all nodes that are in the same level with similar parents directly send the requests to one another.

Other authors [17] proposed a resource management mechanism based on a mobile agent in grid systems. Here, the requester creates a mobile agent, dispatching it to a desired remote machine in the network. The search protocols are defined in the agent with the pertaining code of the remote machine.

In another paper, an exchange protocol based on multiple factors was proposed for resource discovery in the grid [18]. In this mechanism, three types of factors (user, provider, and broker) are used. Each broker factor is connected to the user and provider factors via connection algorithm. This connection mainly consists of four stages: selection, evaluation, filtering, and recommendation.

III. SHUFFLED FROG LEAPING (SLFA)

Nature is a great, inspiring source with a dynamic, abundant exhibition of phenomena for solving highly complex and difficult problems in different disciplines. Meta-heuristic algorithms inspired by nature are smart technologies mimicking nature for solving optimization problems and developing new methods in computer calculations. SFLA is one of the nature-inspired algorithms benefiting from the advantages of the mimetic genetics based algorithm And Particle Swarm Optimization (PSO) algorithm [19]. The effectiveness of SFLA in various optimization problems such as clustering, recommending system, series, etc. is obvious [19-23].

In SFLA, the population consists of a group of frogs searching for food in a pool. Food searching consists of two processes: the inter-group relation of frogs for location discovery, and the intra-group of frogs that belong to different memeplexes of general evolution.

Table 1
Conventional SFLA Pseudo-code [19]

Conventional SFLA
1) Set the dimensions of frogs to d.
2) Initialize the population (P) of frogs (solutions) with random positions. For each frog, compute fitness
3) Sort the population P in descending order of their fitness
*4) Determine the fitness of global best frog (Xg**b**) as fgb*
5) Divide P into m memeplexes
6) For each memeplex m
*a) Determine the fitness of local best(Xl**b**)frog as flb and the fitness of local worst(X**w**) frogs as fw*
b) Try to improve the position of worst frog using Eq. (2) with respect to the local best frog. If fitness improves, update the position of the worst frog.
c) Else
*d) Try to improve the worst frog position using Eq. (4) with respect to the global best frog (X**gb**). If position improves, update the position of the worst frog*
e) Else
f) Replace the worst solution with a new randomly generated solution
7) End
8) Combine the evolved memeplexes
9) Sort the population P in descending order of their fitness
10) Check if termination is true
11) End

Table 1 shows Conventional SFLA Pseudo-code. In SFLA, the initial population (P) is randomly generated for the number (n) of frogs (F_i). After evaluating the initial solutions for suitability, the total population is arranged based on the descending order of their proportionality. Thus, all frogs are distributed to memeplexes (m):

$$MP^d = \left\{ F_k^d \mid F_k^d = F_{d-m(k-1)}, k = 1,2,....,n \right\} d = 1,2,...m \quad (1)$$

In each memeplex, the position of the words solution (X_w) is improved using the proportions and according to the best solutions (X_{lb}), as in Eq. (2) & (3).

$$Change \text{ in } frog\ position(D_d) = rand() \ x\ (X_{lb} X_w) \quad (2)$$

$$New\ Position\ X_w = X_w + D_d; \quad (3)$$

If the position of the worst solution improves, it can replace any other worse solution in order to improve its position according to the best position of the frog. This process is performed according to Eq. (4) & (5):

$$Change \text{ in } frog\ position(D_d) = rand() \ x\ (X_{lb} X_w) \quad (4)$$

$$New\ Position\ X_w = X_w + D_d \quad (5)$$

IV. METHODOLOGY

As stated before, finding the resource that matches to the user's request and that minimizes query traffic is one of the important issues in grid systems. Thus, in this section, we present a method for resource discovery in grid systems that are inspired by SFLA. In this paper, the Time to Live (TTL) factor is used as a proper indicator for the lifetime of the query, since this indicator – denoting the number of hops occurring during the resource search process through the network nodes – is known as the most suitable indicator in non-structured resource discovery algorithms.

The task starts when a query is created. Initially, it verifies whether the node creating the query has the respective resource. If yes, the task is finished, with the node using the resource in its own possession. Otherwise, the node sends a request to its neighbors. The set of nodes targeted by the request form the *V* vector ($< V_0, V_1, ..., V_{n-1} >$), which is shown as 0 and 1. For instance, if the vector is (*0,1,1,0,1,1,0*), it indicates a direct link between the node creating or sending the request and the destination node. If the neighbors targeted by request possess the requested resource, they send a message to announce that they can provide it for the origin node and the task is completed. Otherwise, the proposed algorithm that is based on the SFLA is used for calculating the best meme.

In the proposed method, at each stage, when the node receives a request, it initially looks up for it in the local resources. If the resource is found, a message is sent to the origin node and the task is completed. Otherwise, each node deduces one unit of the TTL, and in case the request is still valid, it sends the request to its neighbors. Figure 1 shows the stages of the proposed algorithm for resource discovery in grid systems.

As stated before, this algorithm consists of the two phases of "global discovery" and "local discovery". In the global discovery phase, a random population is generated and each of the produced memes is evaluated using Eq. (6):

$$fitness = \frac{1}{(\alpha \times TQF)} \times \frac{1}{(\beta \times MFR)} \quad (6)$$

α And ***β*** are the factors for the parameters "total query forward time" (TQF) and "mean fault rate" (MFR). Their values are obtained experimentally.

TQF is the sum of "each query forward time" (EQFT). The queries are sent by the sender to the neighboring nodes. The objective of TQF is to select the meme with the least delay for query forwarding. TQF is calculated using Eq. (7).

$$TQF = \sum_{i \in X} EQFT_i \quad (Where\ X = \{ i \mid v_i = 1\ for\ 0 \le i \le n-1 \}) \quad (7)$$

The MFR factor is defined as the mean value of query fault rates. It is obtained from the history of previous queries. MFR is calculated using Eq. (8).

$$MFR = \sum_{i \in X} EQFR_i \Big/ \sum_{i \in X} QN_i \quad (Where\ X = \{ i \mid v_i = 1\ for\ 0 \le i \le n-1 \}) \quad (8)$$

In this **e**quation, each query fault rate (EQFR) denotes the fault rate for each query. Query number (QN) is the number of queries sent by the sender to the neighbor nodes that are equal to 1 in the meme. The purpose of this parameter is to select the meme that yields the highest rate of incident.

If each of the TQF and MFR parameters has a small value, the evaluation function grows in value. As a result, TQF and MFR must have the least latency and the least fault rate, respectively.

After evaluating the frogs, they are distributed among a number of parallel populations. Each of these populations must be able to evolve separately. In order to prevent being trapped in the local optimum, a number of the frogs (Q) who possess better ideas are selected for continuing the evolution process. This stage is the local discovery stage. After that, it is attempted for the idea of the worst frog to close its distance to that of the best frog. Then, the Q number of frogs are returned to their respective memeplexes and subjected to evaluation. At this stage, the efficiency of the frog changes and it is compared with its previous efficiency. If no progress is observed, continuing the evolution process is futile and a new frog with a random idea is generated to replace the former. After that, the memeplexes are returned to the original array and they are arranged in descending order. This process iterates multiple times until the efficiency of the best frog remains unchanged for several stages. Finally, the meme with the highest evaluation (pX) is selected and the query is sent to the nodes with a value of 1 in that vector.

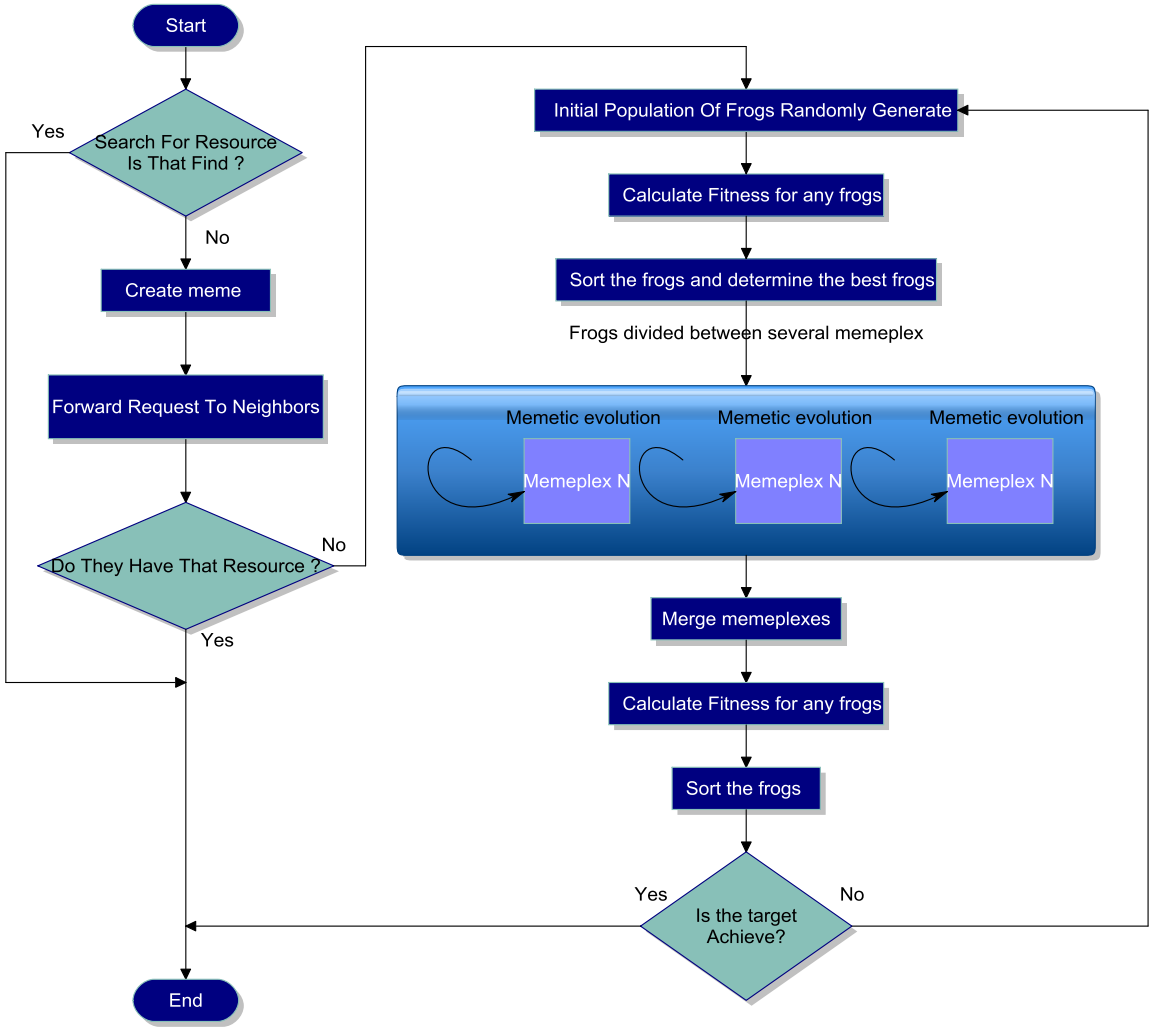

Figure 1: Stages of the proposed algorithm

V. Results and Findings

To evaluate the proposed method, firstly, a number of random nodes were generated and the resources were distributed among them.

Table 2
Controlling parameters for the algorithms

Algorithms	Population Size	Number of Resource	Control Parameters
GA	30	8	α =1 β =2
SFLA	30	8	α =1 β =2 No of memeplex = 4
DEA	30	8	α =1 β =2

In order for a node to have a resource, the respective field should be set to 1. A value of 0 indicates the inaccessibility of the respective resource at the time. Table 2 shows the parameters used for controlling the behavior of the utilized algorithms.

In the first test, SFLA was compared to the methods based on GA and DEA according to the variations in evaluation function. Table 3 and Figure 2 present the variations in the evolution function under different iterations for each method.

Table 3
Variations in the Evaluation Function

Iteration	DEA	GA	SFLA
1	0.1430	0.166	0.285
2	0.1430	0.166	0.309
3	0.1430	0.166	0.312
4	0.1847	0.250	0.325
5	0.2052	0.250	0.330
6	0.2337	0.250	0.333
7	0.2337	0.250	0.366
8	0.3342	0.500	0.395
9	0.3557	0.500	0.476
10	0.3557	0.500	0.595

Table 3, Figure 2 and the comparison of the best and worst values of the variation in the evaluation function indicate a relatively close performance by the three algorithms. During the iterations of 8 and 9, the GA has a better performance, but the results in this test indicate that SFLA is a more suitable method.

However, the results of the solely variations in the evaluation function cannot serve as a proper criterion for comparing the three optimization algorithms. Thus, another criterion, which considers the number of messages sent in

the network or in other words, the network traffic was used. The results of 10 executions of the optimization algorithm with 500 iterations indicated the superiority of the GA; however, for the iteration counts of 1000, 1500, and 2000, the proposed method delivered superior performance in comparison with the GA and DEA. Table 4 and Figure 3 show the results of the proposed method, GA and DEA based on network traffic.

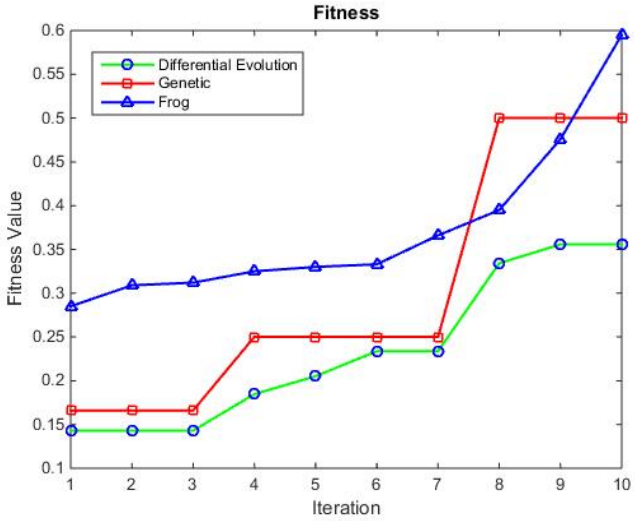

Figure 2: Variations in the evaluation function

Table 4
Network Traffic

Iteration	DEA	GA	SFLA
500	2634	2378	2450
1000	5013	4779	4500
1500	7460	7157	6900
2000	9854	9426	8853

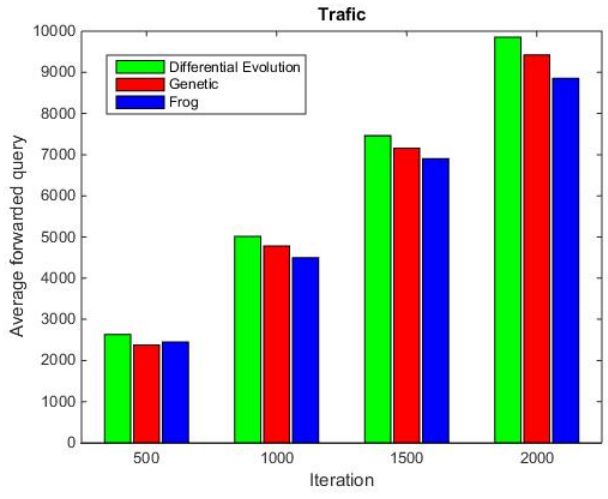

Figure 3: Network traffic

Table 4 and Figure 3 indicate that SFLA except the iteration count of 500 have better performance.

In addition to the variations in the evaluation function and the network traffic, the time required for executing the algorithms and the return of the optimal response is one of the most important criteria that can be used for evaluating and comparing the proposed method and the GA. It is because time is always an important factor in comparing algorithms. Table 5 and Figure 4 compare the methods based on their execution time.

Table 5
Executing time for Each Algorithm

Iteration	DEA	GA	SFLA
500	0.071	0.06	0.009
1000	0.135	0.12	0.018
1500	0.224	0.178	0.026
2000	0.283	0.25	0.039

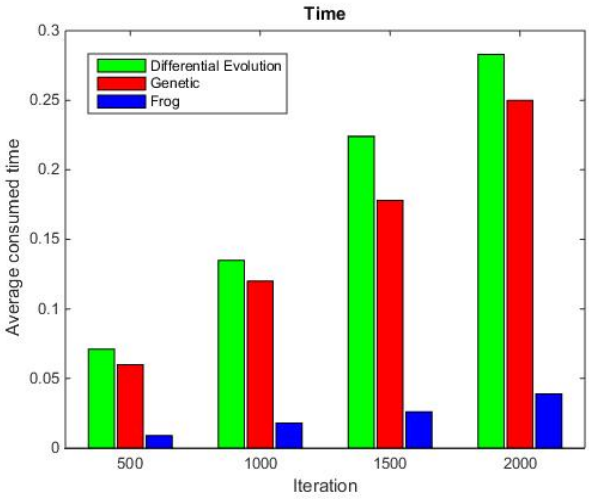

Figure 4: Executing time for each algorithm

As seen in Table 5 and Figure 4, the proposed method delivers significantly superior performance based on execution time. This is important in resource discovery as time is always an important factor in comparing search algorithms.

VI. Conclusion

In grid systems, the user can use the resources of other machines without getting involved in the detailed aspects of addressing. Thus, the discovery of suitable resources for the user's request is one of the important issues in grid systems. For this reason, this paper presented a method based on SFLA in order to discover resources in grid systems, while using the TTL factor to limit the number of sent queries. Subsequently, the proposed algorithm was compared to the GA and DEA based on the value of evolution function, network traffic and execution time. Whilst the proposed method delivered lower performance in some tests, the overall results indicate better performance regarding the speed and network traffic. It is noteworthy that in this paper this method was implemented in inconsistent environments, so for the future research, it can be expanded in consistent environments.

References

[1] Balasangameshwara, J. & N. Raju, *A hybrid policy for fault tolerant load balancing in grid computing environments.* Journal of Network and computer Applications, 2012. 35(1): pp. 412-422.

[2] Foster, I. & C. Kesselman, *The Grid 2: Blueprint for a new computing infrastructure*. 2003: Elsevier.

[3] Ennahbaoui, M. & H. Idrissi, *Zero-Knowledge Authentication and Intrusion Detection System for Grid Computing Security*, in *Information Innovation Technology in Smart Cities*. 2018, Springer. pp. 199-212.

[4] Reddy, T.S.K., D.N. Raju, P.R. Kumar & S.R. Kumar, *Power Aware-Based Workflow Model of Grid Computing Using Ant-Based Heuristic Approach*, in *Big Data Analytics*. 2018, Springer. pp. 175-184.

[5. Benmalek, M., Y. Challal, A. Derhab & A. Bouabdallah, *VerSAMI: Versatile and Scalable key management for Smart Grid AMI systems.* Computer Networks, 2018, 132, pp. 161-179.

[6] Iamnitchi, A. & I. Foster, *A peer-to-peer approach to resource location in grid environments*, in *Grid resource management*. 2004, Springer. pp. 413-429.

[7] Navimipour, N.J., A.M. Rahmani, A.H. Navin & M. Hosseinzadeh, *Resource discovery mechanisms in grid systems: A survey*. Journal of Network and Computer Applications, 2014. 41: pp. 389-410.

[8] Noghabi, H.B., A.S. Ismail, A.A. Ahmed & M. Khodaei. *An Optimized Search Algorithm for Resource Discovery in Peer to Peer Grid*. in *Informatics and Computational Intelligence (ICI), 2011 First International Conference on*. 2011. IEEE, pp. 21-24.

[9] Deng, Y., F. Wang & A. Ciura, *Ant colony optimization inspired resource discovery in P2P Grid systems.* The Journal of Supercomputing, 2009. 49(1): pp. 4-21.

[10] Trunfio, P., D. Talia, H. Papadakis, P. Fragopoulou, M. Mordacchini, M. Pennanen, K. Popov, V. Vlassov & S. Haridi, *Peer-to-Peer resource discovery in Grids: Models and systems.* Future Generation Computer Systems, 2007. 23(7): pp. 864-878.

[11] Torkestani, J.A., *A distributed resource discovery algorithm for P2P grids.* Journal of Network and Computer Applications, 2012. 35(6): pp. 2028-2036.

[12] Yin, Y., H. Cui & X. Chen, *The grid resource discovery method based on hierarchical model*. Information Technology Journal, 2007. 6(7): pp. 1090-1094.

[13] Brocco, A., A. Malatras & B. Hirsbrunner, *Enabling efficient information discovery in a self-structured grid.* Future Generation Computer Systems, 2010. 26(6): pp. 838-846.

[14] Mastroianni, C., D. Talia & O. Verta, *A super-peer model for resource discovery services in large-scale grids.* Future Generation Computer Systems, 2005. 21(8): pp. 1235-1248.

[15] Papadakis, H., P. Trunfio, D. Talia & P. Fragopoulou, *Design and implementation of a hybrid p2p-based grid resource discovery system*, in *Making Grids Work*. 2008, Springer. pp. 89-101.

[16] Ebadi, S. & L.M. Khanli, *A new distributed and hierarchical mechanism for service discovery in a grid environment.* Future Generation Computer Systems, 2011. 27(6): pp. 836-842.

[17] Zhao, C., J. Yu & B. Chai. *A study on mobile agent based resource management in grid*. in *International Conference on Knowledge-Based and Intelligent Information and Engineering Systems*. 2007. Springer.

[18] Kang, J. & K.M. Sim, *A multiagent brokering protocol for supporting Grid resource discovery.* Applied Intelligence, 2012. 37(4): pp. 527-542.

[19] Kaur, P. & S. Mehta, *Resource provisioning and work flow scheduling in clouds using augmented Shuffled Frog Leaping Algorithm.* Journal of Parallel and Distributed Computing, 2017. 101: pp. 41-50.

[20] Prakash, D., A. Tripathi & T.K. Sharma, *Application of Shuffled Frog-Leaping Algorithm in Regional Air Pollution Control*, in *Soft Computing: Theories and Applications*. 2018, Springer. pp. 397-403.

[21] Tong, X., Y. Ji, J. Lin, J. Zhu, F. Sun, Y. Zhong, Y. Yang & X. Zhu, *Cooperative spectrum sensing based on a modified shuffled frog leaping algorithm in 5G network.* Physical Communication, 2017. 25: pp. 438-444.

[22] Sharma, T.K. & M. Pant, *Opposition-Based Learning Embedded Shuffled Frog-Leaping Algorithm*, in *Soft Computing: Theories and Applications*. 2018, Springer. pp. 853-861.

[23] Sarathambekai, S. & K. Umamaheswari, *Performance comparison of discrete particle swarm optimisation and shuffled frog leaping algorithm in multiprocessor task scheduling problem.* International Journal of Advanced Intelligence Paradigms, 2017. **9**(2-3): pp. 139-163.