SB-PSO based Secure Moving Average Time-based Fuzzy Resource Provisioning Approach (SBPSO-MATFRPA) with RSA

Sweta Dey
Computer Science & Engg. Dept.
Indian Institute of Technology (ISM), Dhanbad

Piyush Kumar Garg
Computer Science & Engg. Dept.
Indian Institute of Technology (ISM), Dhanbad

ABSTRACT
SB-PSO and RSA based environment consists a lots of end user's requests for resources simultaneously or sequentially in a dynamic environment and it is a big challenge for this environment. In this paper we propose a method: combination of both SB-PSO and Fuzzy Logic with RSA that allocates requested resources by the end user dynamically so that the available resources are fully utilized in an efficient manner. Here the Monitoring components are continuously monitored the requested resources and allocates them accordingly. Here the incoming requests are grouped together and satisfied in such a way that the maximum numbers of available resources are provisioned appropriately and our proposed approach is efficiently measured by finding the performance of resource allocation.

General Terms
Resource provisioning, fuzzy logic, SB-PSO, RSA.

Keywords
Resource provisioning, RSA, FCFS algorithm, Round-Robin algorithm, Throttled algorithm, fuzzy logic, SB-PSO, Cloud.

1. INTRODUCTION
RSA is an asymmetric (or public key) cryptographic algorithm which was invented by Rivest, Shamir and Adleman of MIT. This secure asymmetric algorithm is mostly used via the internet and supports the encryption technique and digital signature for providing the security. It works by encrypting the user data which must be confidential. Due to this policy of RSA, we use this in our proposed methodology. It is one of the fastest growing sectors for the internet based user friendly technology. It allows end user to store and access data securely over the Internet as per their own requirement by using pay as-you-go and on-demand model instead of using their own computer's hard drive with its fastest data rates and low cost technology. An efficient resource management **[3]** means not only amplify the QoS for the end users, but also deals with reducing the consumption of resources. Cloud is a type of distributed computing system in a virtualized environment where, the resources are dynamically provisioned **[1], [30]**. According to Berkeley "Cloud manages all the applications which are must delivered over the Internet for providing services to the end users for the fulfillment of their demand. The services are themselves referred to as Software-as-a-Service (SaaS)" **[2]**. Resource provisioning and optimization technique both are the heart throb process of cloud computing. Here we choose RSA for security, Fuzzy and SB-PSO for optimization because of their merits like parallel distribution, scalability, easy to realize, robustness, and with high flexibility in dynamic environments. For successfully implementing this various researches are still going on in cloud environment. For example a multi-agent Dynamic Resources Provisioning and Monitoring (DRPM) **[4]** System manages the resources which are provided by the cloud service provider and also satisfying the quality of service (QoS) requirements as per the service-level agreement (SLA). In this paper, A SB-PSO based Secure Moving Average Time-based Fuzzy Resource Provisioning Approach (SBPSO-MATFRPA) with RSA is the algorithm which is proposed for provisioning the resources efficiently so that it provide an efficient fuzzy based resource management technique. Here, Score based PSO and Fuzzy logic is introduced for the system accessibility in between the end user's requirements and resources. Goal of our proposed algorithm is to reduce the total waiting time of the user requirements and producing an efficient resource utilization technique. In this paper work, we focus on developing a secure RSA based methodology; for this we introduced SB-PSO and fuzzy logic based scheme to measure the user's requests for resources and allocate it for maximum utilization **[5]** and **[6]**.

The skeleton of this paper is depicted as section 1 describes Introduction of proposed methodology. In section 2 describes Problem Formulation. Section 3 describes Proposed Methodology. Section 4 describes various resource provisioning approaches. Section 5 is based on the result in this section we presented our simulated output. Finally, in the section 6 a conclusion is given about the improvement of our proposed methodology and also the future scope is discussed here.

2. PROBLEM FORMULATION
Resource provisioning has some common issues like proper optimization, proper scheduling, over provisioning or under provisioning while allocating the resources according to the fulfillment of the demand of the end users. Apart from this there are lot many challenges are exists in resource allocation **[7]**, **[24]**.To get rid of these issues, in this paper we proposed a scheme of dynamic resource allocation strategy based on the demands of end users, it uses SB-PSO, RSA and fuzzy logic as well as monitoring components for allocating the remaining resources in order to achieve the maximum utilization of resources.

3. PROPOSED METHODOLOGY
Users can send their requests for resources to the Service Provider (SP) simultaneously or sequentially and they can interact with the environment by using Graphical User Interface (GUI). First, the user request for VMs **[8]** is forwarded to the monitoring component of the system. Then these are processed according to the availability. This allocation is done effectively by using RSA, fuzzy logic as well as SB-PSO. Here we use fuzzy logic for measuring the

amount of requested resources by the end user, RSA for the confidentiality of the resources and SB-PSO for optimization. Inputs of the system are the total number of requested resources which are coming from the end users and finally the resource utilization is calculated by using a set of moving average time based fuzzy rules and SB-PSO. Calculation of resource utilization for a single user is done by using defuzzification method and SB-PSO. Here SB-PSO is actually finds the available resources and checks for the assignment of the remaining resources or newly added resources to the cloud environment and also try to fulfil the deadline constraint. For doing this job SB-PSO used to give scores to the VMs. If it found that there are no available resources for assigning then it concludes that all resources can't be provisioned successfully, else it takes the fitness value (basically the total execution time as per the solution) of the solution particle and compares it with the given deadline. If the total execution time is greater than the deadline, then it concludes that all resources can be provisioned successfully, but they can't be completed in the given deadline, else it concludes that all resources can be provisioned successfully and they can be completed in the given deadline and solution can be found. After finding the utilization bounds of various server CPUs, we analyze the final score. This factor helps the SP to manage the demands of the users. Lastly, RSA provides the security to protect the system from attacker.

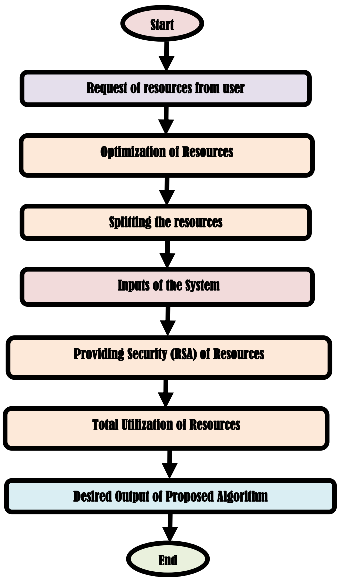

Fig 1: Flow Chart of SBPSO-MATFRPA

3.1 Design of the System

The design of the system is depicted in figure2. It includes the user's requests for resources in the cloud environment. Graphical User Interface (GUI) acts as an interface in between the user and the monitoring component by receiving the request of resources from the end users and then these requests are forwarding to the SB-PSO for getting their respective scores. After completion of these procedures acknowledgement is sent to the monitoring component through a message and it acts as an interface in between the end users and resources.

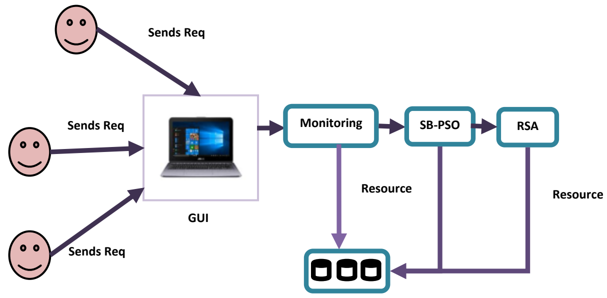

Fig 2: System Architecture

3.2 Monitoring Component

Monitoring components are receiving the user's requests for resources in form of message. Then this message gets splited in various resource categories such as number of processors, memory, RAM, CPU etc. After that these messages are trying to communicate with a cloud repository where actually the resources are stored for further use. Now when all these process are completed then it sends a reply message for indicating the successful completion of resource allocation and it sends a refusal message for reverse case.

3.3 Cloud Repository

It contains the available resources that will be allocated according to the demand of the end users. These required resources are retrieved by the monitoring component at the beginning of the process.

3.4 Score based Particle Swarm Optimization

Score based Particle swarm optimization (SB-PSO) **[9]** is a swarm intelligence paradigm, has been designed to explore and exploit the search space for finding an optimal solution in an efficient manner. PSO consists of a population of particles *Zp*, referred as a swarm. Each particle provides a candidate solution. A particle P_i, $1 \leq i \leq Zp$ has position Xi,d and velocity $V_{i,d}$, $1 \leq d \leq D$ in the d^{th} dimension of the solution space. A fitness function is used to evaluate the quality of the predicted particle. In the initialization process of PSO, each particle is assigned with a random position and velocity to move in a predefined solution space. During each generation each particle finds its own best solution, i.e. personal best represents as $Pbest_i$ and also the overall particles best is global best referred as *Gbest*. To reach the global best solution, it uses $Pbest_i$ and *Gbest* to update the velocity $V_{i,d}$ and position $X_{i,d}$ using the following equations.

$$V_{i,d}(t+1) = w \times V_{i,d}(t) + c_1 \times \chi_1 \times (X_{Pbest_{i,d}} - X_{i,d}) + c_2 \times \chi_2 \times (X_{Gbest_{i,d}} - X_{i,d}) \quad (1)$$

$$X_{i,d}(t+1) = X_{i,d}(t) + V_{i,d}(t+1) \quad (2)$$

Where $0 < \omega < 1$ is the inertia weight, $c1, c2, 0 \leq c1, c2 \leq 2$ are the acceleration coefficients and X_1, X_2 ,$0 < X_1, X_2 < 1$ are the randomly generated values. The updating process is repeated until to reach an acceptable value of *Gbest*. After getting new updated position, the particle evaluates the fitness function and updates $Pbest_i$ as well as *Gbest* as follows.

$$Pbest_i = \begin{cases} P_i, & \text{if } (Fitness(P_i) < Fitness(Pbest_i)) \\ Pbest_i, & otherwise \end{cases} \quad (3)$$

$$Gbest = \begin{cases} Pbest_i, & \text{if } (Fitness(Pbest_i) < Fitness(Gbest)) \\ Gbest, & otherwise \end{cases} \tag{4}$$

The moment of a particle is represented in **Fig. 3.** The position vector and the velocity vector of each particle are set randomly within the search space **[10],[11],[12],[13]**.

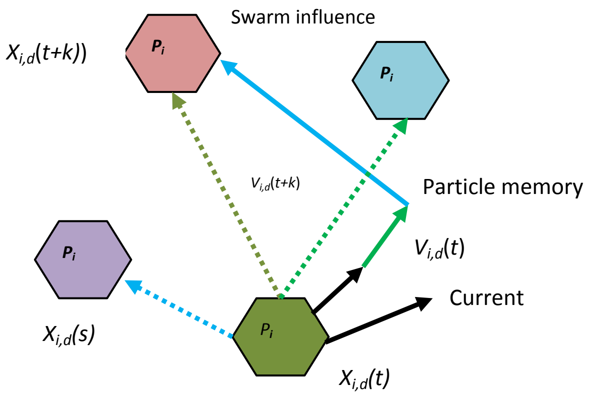

Fig 3: Moment of a particle in the search space

3.5 Rivest, Shamir and Adlenman (RSA)

RSA represents the name of its investors and it is one of the asymmetric-key cryptosystem. Inclusion of digital signature and encryption with it provides security to our methodology. For providing security it works on the two large prime number's for mathematical computation. In this asymmetric key algorithm the encryption and decryption key are totally different. The encryption key is public and the decryption key remains secret. It's really hard to view the exact required resources from the resource pool. Bulk encryption and decryption operations are performed at a higher speed when RSA sends encrypted keys of public key cryptography. It is also an algorithm which is protected from the brute force attack **[31]**, **[32]**.

4. ALGORITHMS USED

4.1. FCFS (First Come First Serve)

It is one of the simplest algorithms **[33]**, **[34]**, **[35]** which are used for processing the end user's request. Here, lots of requests are monitored coming from lots of users and the requested resources are allocated according to their entry order to the system.

4.2. Round-Robin

It is a kind of simplest scheduling technique which mainly deals with time quantum and here this time quantum will fixed for each and every job. It follows a ring architecture for gathering all the jobs from waiting queue and execute them by following fixed time quantum for every job **[14], [15], [16], [17]** i.e. every node is given a fixed time quantum for completing its execution but when the task or job is large then it will take long time to complete the entire task **[18]** i.e. without completion of the entire job it will goes to the next node and so on. It is less complex rather than other resource provisioning algorithms [19], [20], [21], [26], [27], [28], [29].

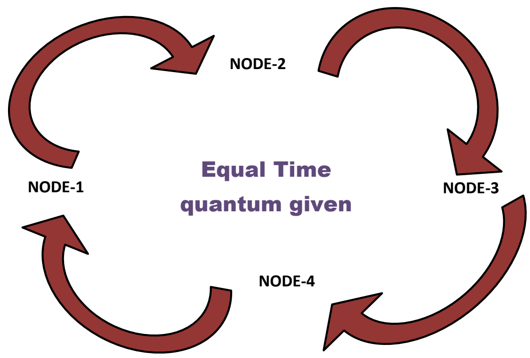

Fig 4: Block Diagram of RR Algorithm [25]

4.3 Throttled

By this algorithm a pre-defined suitable number of requests are allocated to a VM at any given period of time. But, if more requests are present there then finds the available VMs at that respective data center and allocate them until the availability of VMs. It is a better approach as compared to the previous one i.e. RR Algorithm **[22], [23]**.

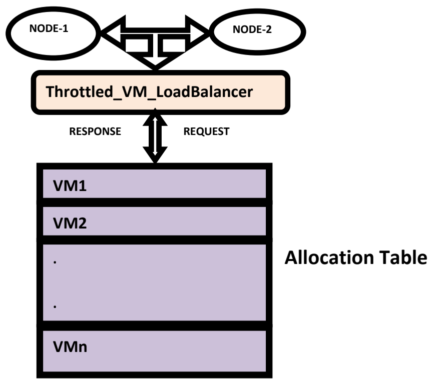

Fig 5: .Block Diagram of Throttled Algorithm [25]

5. EXPERIMENT AND RESULT

The proposed algorithm is implemented in java and simulation is done through Cloudsim toolkit and Cloud Analyst Simulator. We create cloudlets, VMs, Datacentres with specific configuration which satisfies our proposed methodology. The toolkit and simulator provides the facilities to create the related setup for experimental analysis. After the submission of cloudlet requests for the participating VMs, the resource allocation policy has been applied. It shows how the resources are provisioned to the VMs of the datacentres. After the execution scenario, we made comparisons: response time and data centre processing time of the existing algorithms is compared with the proposed algorithm in **Table 1**, statistical comparison among all the resource allocation algorithms is depicted in **Table 2.** Therefore, it presents the comparison scenario of HMAT-FRS. **Table 3**, **Table 4**, **Table 5** and **Table 6** present the response time of proposed and other existing resource allocation approaches.

Table 1: Response Time and Datacenter Processing Time of Various Resource Provisioning Approaches

		Avg (ms)	Min (ms)	Max (ms)
SBPSOMAT-FRPA (Proposed)	Overall Resp. Time	305.65	256.2	355.2
	Datacenter Processing Time	0.50	0.20	0.18
FCFS [33], [34], [35]	Overall Resp. Time	319.1	274.2	379.3
	Datacenter Processing Time	0.45	0.20	0.79
Round-Robin [11]	Overall Resp. Time	312.1	247.5	356.2
	Datacenter Processing Time	0.45	0.12	0.77
Throttled [12]	Overall Resp. Time	312.5	247.5	356.1
	Datacenter Processing time	0.37	0.05	0.75

Table 2: Statistical Comparison among Various Resource Provisioning Approaches

Recourse Provisioning Algorithms	Resp. Time	Throughput	Performance	Resource Utilization
FCFS [33], [34], [35]	Slow	Low	Low	Good
Round Robin[11]	Fast	Low	Low	Good
Throttle[12]	Fast	Low	High	Good
SBPSOMAT-FRPA (Proposed)	Very Fast	High	High	High

Table 3: Response Time of Proposed Approache

UserBase (UB)	Avg (ms)	Min (ms)	Max (ms)
UB1	314.35	271.13	376.15
UB2	289.16	263.61	353.64
UB3	315.88	272.59	368.64
UB4	314.08	253.11	362.64
UB5	287.68	257.18	364.15

Table 4: Response Time of FCFS Algorithm

UserBase (UB)	Avg (ms)	Min (ms)	Max (ms)
UB1	317.35	273.13	378.15
UB2	311.16	265.61	355.64
UB3	317.88	274.59	369.64
UB4	315.08	255.11	364.64
UB5	297.68	249.18	366.15

Table 5: Response Time of Round-Robin Algorithm

UserBase (UB)	Avg (ms)	Min (ms)	Max (ms)
UB1	313.67	271.13	376.15
UB2	311.15	263.64	364.15
UB3	316.09	273.59	368.64
UB4	312.89	253.11	362.64
UB5	297.42	257.18	364.07

Table 6: Response Time of Throttled Algorithm

UserBase (UB)	Avg (ms)	Min (ms)	Max (ms)
UB1	314.87	271.13	376.15
UB2	289.14	247.18	344.64
UB3	316.28	272.59	368.64
UB4	312.07	253.14	362.64
UB5	299.86	257.62	364.15

6. CONCLUSION AND FUTURE WORK

In this paper our proposed methodology makes the resource provision approach efficiently and effectively in the cloud environment. This aim of this technique will fulfilled successfully by our experiment and this can be done by more than 50 times and all the time it gives the same results towards the goal. In future work, we plan to explore: integration of the proposed approach with Rough Algorithm, using the proposed approach for more efficient for cloud applications in cloud environment.

7. ACKNOWLEDGEMENT

We would like to acknowledge "International Journal of Computer Application" society for giving us an opportunity to present our research paper.

8. REFERENCES

[1] Jing Bi, Zhiliang Zhu, Ruixiong Tian, Qingbo Wang, "Dynamic Provisioning Modeling for Virtualized Multi-tier Applications in Cloud Data Center", Proceedings of 2010 IEEE 3rd International Conference on Cloud Computing, 978-0-7695-4130-3/10

[2] Fox, A., Griffith, R., Joseph, A., Katz, R., Konwinski, A., Lee, G., & Stoica, I. (2009). Above the clouds: A Berkeley view of cloud computing. Berkeley, Rep. UCB/EECS, 28, 13.

[3] Jiangtao Zhang, Hejiao Huang, Xuan Wang, “ Resource Provisioning Algorithm in Cloud Computing: A Survey” , Proceedings of the Journal of Network and Computer Applications(2016), j.jnca.2015.12.018.

[4] Ghobaei-Arani, M., Jabbehdari, S., & Pourmina, M. A. (2018). An autonomic resource provisioning approach for service-based cloud applications: a hybrid approach. Future Generation Computer Systems, 78, 191-210.

[5] Yu Sun, Jules White, Sean Eade, Douglas C. Schmidt, "ROAR: A QoS-oriented modeling framework for automated cloud resource allocation and optimization," Elsevier The Journal of Network and Computer Applications, Volume: 41, 2014, pp:424–440.

[6] Chunlin Li, La Yuan Li," Optimal resource provisioning for cloud computing environment," Springer The Journal of Super Computing, Volume:62, Issue:2 ,November 2012, pp:989-1022.

[7] Sunilkumar S. Manvi, GopalKrishna Shyam," Resource management for Infrastructure as a Service (IaaS) in cloud computing: A survey," Elsevier Journal of Network and Computer Applications , Volume:41, 2014, pp:424–440.

[8] Weijie Shi, Linquan Zhang, Chuan Wu, Zongpeng Li, and Francis C. M. Lau, " An Online Auction Framework for Dynamic Resource Provisioning in Cloud Computing, " IEEE/ACM Transactions on Networking, Volume:24 Issue:4,August 2016 , pp-2060-2073.

[9] Kennedy, J. (2010). Particle swarm optimization. In Encyclopedia of Machine Learning (pp. 760-766). Springer US.

[10] Clerc, M., & Kennedy, J. (2002). The particle swarm-explosion, stability, and convergence in a multidimensional complex space. Evolutionary Computation, IEEE Transactions on, 6(1), 58-73.

[11] Zhang, R. (2011). A particle swarm optimization algorithm based on local perturbations for the job shop scheduling problem. International Journal of Advancements in Computing Technology, 3(4), 256-264.

[12] Yang, C. H., Tsai, S. W., Chuang, L. Y., & Yang, C. H. (2011). A modified particle swarm optimization for global optimization. IJACT: International Journal of Advancements in Computing Technology, 3(7), 169-189.

[13] Yang, J., & Yang, J. (2011). Intelligence optimization algorithms: A survey. IJACT: International Journal of Advancements in Computing Technology, 3(4), 144-152.

[14] Samal, P., & Mishra, P. (2013). Analysis of variants in Round Robin Algorithms for load balancing in Cloud Computing. International Journal of computer science and Information Technologies, 4(3), 416-419.

[15] Sran, N., & Kaur, N. (2013). Comparative analysis of existing load balancing techniques in cloud computing. International Journal of Engineering Science Invention, 2(1), 60-63.

[16] Subashini, S., & Kavitha, V. (2011). A survey on security issues in service delivery models of cloud computing. Journal of network and computer applications, 34(1), 1-11.

[17] Megharaj, G., & Mohan, K. G. A Survey on Load Balancing Techniques in Cloud Computing.

[18] Kansal, N. J., & Chana, I. (2012). Existing load balancing techniques in cloud computing: a systematic review. Journal of Information Systems and Communication, 3(1), 87.

[19] Chaczko, Z., Mahadevan, V., Aslanzadeh, S., & Mcdermid, C. (2011, September). Availability and load balancing in cloud computing. In International Conference on Computer and Software Modeling, Singapore (Vol. 14).

[20] Roy, A., & Dutta, D. (2013). Dynamic load balancing: improve efficiency in cloud computing. International Journal of Emerging Research in Management Technology, 78-82.

[21] Kazem, A. A. P., Rahmani, A. M., & Aghdam, H. H. (2008, August). A modified simulated annealing algorithm for static task scheduling in grid computing. In Computer Science and Information Technology, 2008. ICCSIT'08. International Conference on (pp. 623-627). IEEE.

[22] LD, D. B., & Krishna, P. V. (2013). Honey bee behavior inspired load balancing of tasks in cloud computing environments. Applied Soft Computing, 13(5), 2292-2303.

[23] Kawser, M. T., Hasib, M. A. B., Hasin, A. R., Sadik, A. M., & Razu, I. K. (2012). Performance comparison between round robin and proportional fair scheduling methods for LTE. International Journal of Information and Electronics Engineering, 2(5), 678.

[24] Dr.S.Padmavathi, N.Soundarya, P.K.Soniha, S.Srimathi (2017). Dynamic Resource Provisioning and Monitoring for Cloud Computing. International Conference on Intelligent Techniques in Control, Optimization and Signal Processing, 978-1-5090-4778 9/17.

[25] Kaneria, O., & Banyal, R. K. (2016, November). Analysis and improvement of load balancing in Cloud Computing. In ICT in Business Industry & Government (ICTBIG), International Conference on (pp. 1-5). IEEE.

[26] Lee, R., & Jeng, B. (2011, October). Load-balancing tactics in cloud. In Cyber-Enabled Distributed Computing and Knowledge Discovery (CyberC), 2011 International Conference on (pp. 447-454). IEEE.

[27] Lorpunmanee, S., Sap, M. N., Abdullah, A. H., & Chompoo-inwai, C. (2007). An ant colony optimization for dynamic job scheduling in grid environment. International Journal of Computer and Information Science and Engineering, 1(4), 207-214.

[28] Kaur, J., & Kinger, S. (2014). A survey on load balancing techniques in cloud computing. International Journal of Science and Research (IJSR), 3(6), 2662-2665.

[29] Zhou, M., Zhang, R., Zeng, D., & Qian, W. (2010, October). Services in the cloud computing era: A survey. In Universal Communication Symposium (IUCS), 2010 4th Internationa l(pp. 40-46). IEEE.

[30] Buyya. R., Yeo, C. S., Venugopal, S., Broberg, J., & Brandic, I. (2009). Cloud computing and emerging IT platforms: Vision, hype, and reality for delivering

computing as the 5th utility. Future Generation computer systems, 25(6), 599-616.

[31] Bhanot, R., & Hans, R. (2015). A review and comparative analysis of various encryption algorithms. International Journal of Security and Its Applications, 9(4), 289-306.

[32] Longhua, L. (2012, July). A novel design of OTP-based authentication scheme using smart phones and 2-D barcodes for the visually impaired. In Proceedings of the 6th International Conference on Rehabilitation Engineering & Assistive Technology (p. 8). Singapore Therapeutic, Assistive & Rehabilitative Technologies (START) Centre.

[33] Gallager, R. G. (1978, September). Conflict resolution in random access broadcast networks. In Proc. of the AFOSR Workshop in Communication Theory and Applications (pp. 74-76).

[34] Mosely, J. (1979). An efficient contention resolution algorithm for multiple access channels (No. LIDS-TH-918). MASSACHUSETTS INST OF TECH CAMBRIDGE LAB FOR INFORMATION AND DECISION SYSTEMS.

[35] Vvedvenskaya, N.D., and Pinsker, M.S., "The Capacity of FCFS Algorithms," in Proc. 3rd Intern. Seminar on Convolutional Codes and Multiuser Communication, Sochi, U.S.S.R, 1987, pp. 32-34, [in Russian].