Ant colony algorithm for text classification in multicore-multithread environment

Ahmad Nazmi Fadzal, Mazidah Puteh, Nurazzah Abd Rahman
Faculty of Computer and Mathematical Sciences, Universiti Teknologi MARA, Malaysia

Article Info

Article history:

Received Oct 10, 2019
Revised Dec 14, 2019
Accepted Dec 28, 2019

Keywords:

ACO
Ant colony optimization
Multicore-multithread
Swarm Intelligent
Text classification

ABSTRACT

This paper presents about Ant Colony Algorithm (ACO) for Text Classification in Multicore-Multithread Environment in Artificial Intelligent domain. We had develop a software which assimilate concurrency concept to multiple artificial ants. Pheromone in ACO is the main concept used to solve the text classification problem. In regards to its role, pheromone value is changed depending on the solution finding that has been discovered at the pseudo random heuristic attempt in selecting path from text words. However, ACO can take up longer time to process larger training document. Based on the cooperative concept of ants living in colony, the ACO part is examined to work in multicore-multithread environment as to cater additional execution time benefit. In running multicore-multithread environment, the modification aims to make artificial ants actively communicate between multiple physical cores of processor. The execution time reduction is expected to show an improvement without compromising the original classification accuracy by the investment of trading on more processing power. The single and multicore-multithreaded version of ACO was compared statistically by conduction relevant test. It was found that the result shows a positive time reduction improvement.

Corresponding Author:

Ahmad Nazmi Fadzal,
Faculty of Computer and Mathematical Sciences,
Universiti Teknologi MARA,
40450 Shah Alam, Selangor, Malaysia.
Email: ahmadnazmi@uitm.edu.my

1. INTRODUCTION

In a colony of ant, the ant assigned to search for food is able to track and judge a shorter route over time. This behavior can be explained by the use of pheromone left by the ant along traveled path and change of its concentration that produce different strength of smells which can be used as an effective communication. A derivation of that pattern, an algorithm, Ant Colony Optimization (ACO) is researched as a wrapper method which can be used to solve problem with buildup solution. This includes of probabilistic and meta-heuristic usage contribute to solve difficult optimization problem [1-3]. The artificial ant will continuously attempt to increase the quality of solution over repeating algorithm life cycle.

For this paper, ACO is explored to solve text classification problem. The goal is to distribute a set of documents into two distinctive categories which is crucial to put them in organized and structured manner. The main task focused on supervised learning which requires the user to explicitly train the prototype in order to classify test documents after applying the pre-processing part. One of major problems occur in text classification challenge is curse of dimensionality that reduce the effectiveness of algorithm especially the non-statistical approaches when processing data in high-dimensional spaces [4]. Besides that, ambiguous meaning of a term that always happen in a document may abstain a classification model to strive hundred percent classification rates. Based on the situation described, ACO in multicore-multithread environment is introduced to exercise bio-inspired solution with cooperative communication.

The paper uses a version of ACO in that use all terms at once to generate classification rule. The method that is implemented involve in the relationship between each term node visited by artificial ants which interpret as a path length. The algorithm originates from [5] was once applied to solve shortest path problem in Traveling Salesman Problem (TSP). The distances are recalculated as the inversed value of path length to mark out shortest distance between visited terms. To maintain experiments setup credibility, the paper has left related standard setting untouched in order to produce precise and reflect to the adjustment. Term-frequency table approach which allows artificial ants to do real time data analysis will be used in representing the relationship between two unique terms.

In order to cope with a large volume of data processing problem for instance documents stored in data center, multicore-multithreaded computing provides an ample feature of concurrency. Through continuous research, real ants have been discovered to actively interact to each other using pheromone substance regardless of which ants have placed the pheromone. Such feat could be considered as a significant advantage for effective communication in mapping path discovery. ACO interactive feature in multicore-multithreaded environment has potential to find efficient solution in shorter time than a single threaded approach could perform. ACO pheromone concept is suitable to be used in concurrent processing as it involve in simple mathematical calculation heuristic function.

Multicore-multithread that could represent the mechanism of cooperative colony of ants could be adapted with multiple instances of artificial ant. The main objective of the proposed task to be integrated to the original system is to response to ACO running time improvement. In order to achieve the task, artificial ant can be assigned to one thread at a time for each instance of artificial ants. The model proposed try to simulate the real situation of ants' behavior which an ant can work at different speed, interrupted and sharing pheromone value for communication. These situation are identical to problem faced in multicore-multithread environment which a core processor that represent a node could have different specifications, slower processing power and need an effective, real time and rapid communication to another computer node. Different to parallel computing, multithreading is a debatable choice that offer extended capability and flexible to cope with today's multitasking coding. Depicting real pheromone that is open to any other ants that smell it, pheromone in multithreaded is designed to be volatile single copy that shared between multiple instances of artificial ants.

This paper present the text classification using ACO in multicore-multithread environment. The method used is concurrency concept applied to every artificial ant separately using thread function available in C++ language. A test between unthreaded ACO and multithreaded version shows positive result that reduces the classification running time with insignificant trade off to text classification accuracy. Results of the study shown that the concurrency implementation has acceptable improvement to reduce classification time for documents number larger than 50. In this regards, the concept could be useful in real world situation that require to classify high number of documents such as online library and bibliography web server.

a) Text Classification

Text classification task mainly concentrate on attempt to solve most of the related problem of data sparseness. Aliases to text mining, text classification can be perceived as knowledge intensive learning process, which a person able to further access with a document collection over time continuously by using a correct analysis tool [6, 7]. Text documents such as articles and newspaper in their unmodified content cannot be immediately processed by common classifiers and available learning mining methods. In ACO case, the algorithm could use their vector form after converting them into term frequency format in preprocessing step to represent TSP version of related terms attributes. To process different text pattern and main terms, ACO construct the classification rule using IF (term1) AND (term2) THEN (class) statement clause to induced the texts pattern.

b) Ant Colony Algorithm

ACO algorithm has been found by an inspiration experiments ran by [8], that observed from a real colony of ants from Argentine family (Iridomyrmex Humilis). Initially, ants travel the graph with initial solutions [9, 10]. Most of the ants use the shortest branch to fetch food to their nest when observing a few minute of ants behavior by locating a food source to a isolate area connected by a bridge with two path of different length. The experiments deduced that the ants interact and passing route information by placing pheromone along traveled path which known as stigmergy [11, 12]. The most important state in the algorithms is to choose which heuristic to implement to produce solutions.

Ant System (AS) is the first product of ACO algorithm [5, 13] which was successfully solve the well-known TSP [14-18]. Even though AS shows a good result for a wrapper method, it could not surpass the state-of-the-art algorithms to solve TSP problem in term of accuracy. The research problems are fundamentally the same but the algorithm applies in different discipline [19, 20]. Available in different approach, Ant Colony System by [21] has opposite exploits to AS by Nezamabadi-Pour et al approaches [19], as stated by [20], while cAnt-Miner is handling with continuous attributes [22, 23]. As to this research, the algorithm approach adapts TSP solution. It has been experimented that the fundamental difference method is crucial to the respective designed ACO-based algorithms [24].

In subsequent research of Ant-Miner, a research featuring cAnt-Miner (Ant-Miner input with continuous attributes), includes an entropy-based discretization method in order to cope with continuous attributes in a process when discovering a new rule. cAnt-Miner is known to able to create individually separate intervals for continuous attributes "on-the-fly", taking advantage of all continuous attributes information, rather than requiring that a distinctive method be used in a preprocessing step. In order to discover new improvement, many Ant-Miner variations have been proposed which of them unable to differentiate attributes "on-the-fly" (process to discover new rules) as researched in [23, 25]. Other proposed method also has been experimented that integrate the state-of-the-art classification method; Support Vector Machine to ACO as stated in [26].

In the surrounding of real environment, a sign of cooperative manner particularly when foraging for food is shown clearly by multiple of real ant. Similarly, the ACO algorithm also could be further experimented for multiple instances of artificial ants. Since its discovery, there are a number of extensive researches which investigate ACO for parallelism or strategic use of pheromone in concurrency feature [27-30]. One of the researches include in numerical optimization that use multicore coordination to distinguish stigmergy of ant algorithm. Regards to graphical processing units, parallel ACO had been studied to derive effective parallelization strategies. In continuous-attribute classification rule exploration, another research had been conducted to employ numerous pheromones deposit in ant-related algorithm. For real time performance test of multicore arrangements, a specific simulation to run extensive evaluation is developed [31-34].

In the earlier days of computer history, processor used was limited by processor capability. Multithreading comes in handy to address the problem. For a single core computer, multithreading can be done on it but not parallelism [35]. In other word, parallelism needs at least two real processer cores to work. On the other hand, thread switching can be implemented at software level together with minimal instruction set in order to use multithreading on a single machine.

As the ACO algorithm is a derivation from bio-inspired ant community that runs their living in organized life cycle [36, 37], the algorithm theoretically can be explored for coordination potential of concurrency design in multicore-multithread environment. Over a true parallelism, multithreading operation from the multicore-multithread environment can be aligned with several instances of artificial ant. The focused goal for the invented original ACO method to be adjusted to indigenous system is to further enhance ACO execution time. In contemplation to implement out the task, artificial ant can be assigned to one allocated child thread at a time for distinct instance of artificial ants. The indicated idea is suggested in the side to try to reflect the actual situation of real ants' imitation which an ant can advance at autonomous speed, take some rest and deploy pheromone as an apparatus for communication [38, 39]. These features can yield the problem aspect in releasing multiple instances of artificial ant whenever a thread that guide the artificial ant could have exclusive priority, distinctive status and require a potent, real time and rapidly shuffle state information to other child threads. Differentiable from parallel computing, multithreading as a suitable option that overtures expand capability and flexibility to cater the more than one instances of artificial ant. Duplicating real pheromone apparatus that facilitate route evaluation for any other ants that found it, pheromone in multithreading is constitute to be volatile to a single copy which can be analyze simultaneously between multiple threads of artificial ants. An examination between single threaded and multithreaded variant of ACO shows sufficient results that reduces the text classification running time.

In [40], have enhanced the findings to search for classification rules within parallel context. The method discovery is based on a course-grain master slave model of ACO taxonomy hierarchy that was implemented to process the solutions. According to the techniques, better pheromone update is selected from the best colony when a classification rule is discovered. The improvement is further extend with better predictive accuracy by other previous ACO related work "by using efficient communication methods" as claimed by the author. One of the models applies a normal multi-colony approach that communicates through Message Passing Interface. The complementary model is a fine-grain master-slave type that capable to organize pheromone access by using multithreading which the parent or master process act as a global access that solve pheromone sharing problem.

2. RESEARCH METHOD

2.1. Text Classification Framework with ACO

This research adapts two main phases in ACO for text classification in multicore-multithread environment. The first one is preprocessing phase that produces the functioning input for the second phase.

The ACO will handle training document set to examine text classification rules that complement ACO classification model. Filtered data from training documents will be parsed and indexed before frequency of each term is computed.

Text classification setting out organizing list of term onto nodes as it will be used as a graph. Each node communicates to one another at bidirectional relationship. This stage is accustom from TSP. In TSP, cities are presented as nodes which a businessman required to visit all cities exactly once. Identical to this requirement, we instruct the artificial ants to take place as the agent to tour to all cities and calculate shortest distance out of all possible path. The calculation complexity of the problem presented in this scheme is Non-Polynomial Hard type or simply NP-hard. The paper treats the training document into two distinctive classes of positive and negative affinity. It also indicate that the artificial ant require to make a choice between one of the categories every time the ant visit a node. The artificial ant will mark the chosen class by increasing pheromone value based on heuristic calculation of previous visit. Figure 1 expresses the mechanism used by ACO for each term:

Having artificial ants to tour all different path in succession could waste too much time even when run on supercomputer. A technique named Brute Force Algorithm is the one that check every possibility without selecting out the most unlikely result with heuristic function. In example of 60 cities, if a businessman is in request to travel to all cities, the maximum number of possible outcome are 60^{60} = 4.89e+106 unique way. Using ACO method, it can find the most optimal path in less than 500 attempts.

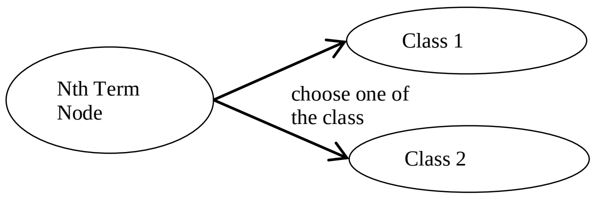

Figure 1. Decision making on term nodes

2.2. ACO Algorithm

2.2.1 ACO Cycle

Iterations or in more specific term, cycle indicate to the continual execution of fundamental algorithm part in ACO repeatedly in order to develop mature heuristic value of pheromone as part as incorporate procedure in wrapper method. Strategically, all of its code occupy in main loop. The following is the pseudo-code that illustrate the part which contribute as the main algorithm:

```
1  procedure ACO_search();
2  while (termination_condition_not_occur)
3        schedule_tasks
4            create_and_assign_ants();
5            update_path_selection();
6            update_measurement();
7            update_pheromone();
8        end schedule_tasks
9     end while
10 end procedure
```

For a single cycle, it refers to a set of path that has been visited and complete pheromone value has been calculated has been calculated exactly once. Subsequently, each nodes in the graph will have pheromone reduction by one percent which depict a constant evaporation phenomenon that happens in real world situation. The progress outset with artificial ants choosing vertices or route and proceed with score calculation on correct selection of provided training classes. At the end of the artificial ants' cycle, pheromones value get renewed using the compute score value from earlier step.

Cycle is significant aspect of ACO as it consist of the maturing process of pheromone value through heuristic calculation over repetitive iteration. Literally, there is no precise number that convenient to set in experiment setting as it all related on the complexity of classification problem. As it differentiate this wrapper method to other methods, there are a few of service task are appointed at the end of every cycle such as correlate anterior nodes, renewing score and creating new artificial ants.

By assigning artificial ants to a new thread, they are dispatched and executed through a loop. Even though the dispatching phase is run in sequential manner, the order of they are released could easily be changed upon executing them in a thread. These things occur because there are times that execution speed occasionally drops because a child thread that manages an instance of artificial ant is short on resource. On operating system situation, executing multiple requests from software application level may take various operations time or typically slower. Assigning some task which has uninterruptible execution block or atomic execution sequence is more difficult and complex to manage. The problem arises when the task requires longer allocation that delays other task from executed timely. Other concern to the problem also is a task can request to be executed in higher priority. Operating system might not be able to fulfill all requests directly because time is immutable and limited resource. The problem will result in holding other task that waiting in a queue that originally require shorter execution time or consume less resource. If the task executed at later time than originally requested, data inconsistency could happen due to incorrect timing of the time function and resource availability issue.

Besides that, time function also need a careful and precisely executed before and after searching phase commence. As many threads will use same code blocks concurrently, micro management for time function is important to prevent incorrect timestamp. Time recording function in programming can be very sensitive, as it record time ticks in nanosecond. This means that we could observe a large time difference when task execution time is moved only a little. The situation can be improved by the operating system by having a task to repeatedly apply for context switching on running thread. Thread context switching give a chance for smaller or lighter task to be selected ahead of waiting queue and also allows larger thread to be broken into smaller parts. However, there is a concern to context switching when it is applied to a high number of threads, especially small and lighter threads. Context switching should not be abused as it incur additional problem such as creation overhead and may result in slow responsiveness to operating system. It is usually applied when handling larger task that requires to be broken into smaller parts so that other smaller task can be executed early.

As artificial ants continuously visiting nodes, they require to exchange information among themselves to effectively share their discovery. Features offered from using multithreading are simpler to use because a child thread that created from a parent thread able to have easier access to their global variable in addition to easily send data through normal function as an instantaneous communication compared to Inter-Process Communication (IPC). This attribute is critical in order to achieve high calculating accuracy of time for measuring operation time of every artificial ant especially when stopping other artificial ant by using global signal variable. Because multithreading may have uneven speed behavior, recording accurate finish time of every thread could be difficult to maintain. This is due to the parent thread are on hiatus while waiting all of child thread to be done from executing assigned tasks. The parent thread also is bound with a contract against to its child threads that is sealed with synchronized code. Most of the time contract is required to ensure that all of created child threads are successfully started and dismissed before getting their total lifespan solution evaluated. If the child threads are not correctly released or muted, they could have run endlessly in their own loop in searching space. Therefore, synchronize mechanism must enforce as a complementary action for signaling stopping condition.

Signaling stopping mechanism require a child to bind a contact its parent thread through global space before acquire a list of child thread. Then the issuer threads will go through a temporary loop to set all of stopping variable of every child thread as stopping mechanism. Every other child thread that read stopping condition need to update a current timestamp as soon as possible after finish their searching loop. After all of artificial ants reach the end of its lifecycle, the parent thread will be wake up from its hiatus as a resulting effect of synchronize contract. Now, parent thread is ready to total up average duration and examine the solution of every artificial ant for current cycle.

3. RESULTS AND ANALYSIS

In order to analyze the proposed ACO algorithm in multicore-multithread, paper has selected The 20 Newsgroups data set [41]. However the textual data for different topic has unequal number of document. Considering the scope of this paper is to normally experiment the attribute of ACO in multicore-multithread time reduction, the number of training document has been balanced and scaled for 50 files addition per run in both category.

An adequate set of ACO run to regulate the most suitable number of artificial ants to run in synchronize manner is performed manually. The run select 100 documents as the input training set for classification and it is run to 16 times with the exact setting with incremental number of concurrent artificial ants. The test conducted on machine with Intel(R) Core(TM) i7-3630M processor that has clock speed of 2.4GHz, 4 core and 8 factory thread. Installed operating system is Windows 7 64bit, Ultimate edition. Maximum number of thread chosen is 16 because the time reduction values marginally decreasing as it exceed the maximum number of factory thread. All of the test execution time after filtering text data were recorded to analyze the fastest time.

For the mentioned test machine attributes, using nine threads concurrently are concluded to be the best time for current test setting. Based on the result, we found that 35.53% time reduction is gained with nine thread used on 100 data set. To complete the experiment, original ACO for text classification has been selected as the opposing experiment comparison to the multicore-multithread version of ACO in aspect of time reduction. The experiment was repeated 10 times (10 fold) to cross validate the average accuracy. Below is the experiment result between ACO in multicore-multithread environment time reduction:

Table 1 shows the time reduction comparison between two version of single threaded and multicore-multithreaded ACO. The least time improvement can be observed on the least text files used which is 14.28%. The multicore-multithreaded ACO has insignificant 0.03 second faster than the other test. Based on this analysis, it can be stated that the proposed method provide improved time reduction. Time reduction value is gained as the largest benefit on 200 text files run. By a 59.85% better than its comparable unthreaded ACO, ACO in multicore-multithread record 3.4 seconds less from 5.68 seconds in the last run. Based on the analysis, the technique proposed is recommended. Therefore, the classification time problem has been addressed.

Table 1. Time Reduction in Seconds between Single Threaded and Multicore-Multithreaded ACO

Number of File	50	100	150	200
Unthreaded ACO (seconds)	0.21	0.76	1.71	5.68
Multicore-Multithread ACO (seconds)	0.18	0.49	1.27	2.28
Time Reduction	14.28%	35.52%	25.73%	59.85%

4. CONCLUSION

ACO in multicore-multithreaded environment is found to be less suitable to be used when it is used on smaller number of training document. This situation could happen because there is overhead to create and initialize multiple artificial ant repeatedly each cycle before they are ready to be used. Other factor that might affect this result is how operating system handles multiple request of thread and speed of memory allocation for different number of request. Processor architecture to handle multiple core and threads could contribute random benefit time reduction as the architecture has its own optimal threading operation between cores. In other situation, ACO in multicore-multithread environment has better time reduction benefit when used for larger number of training document. The benefit gained from multithreading could outweigh the overhead cost to operate multicore-multithread. Therefore it is recommended to consider the proposed method for larger corpus that could lead to long time consuming text classification. As a future work, ACO in multicore-multithread environment can be further improved by distributing its main operation in cloud environment. For easier connection, it also could be adapted to local area network that ideal to be used on private network in an organization. Training document should be mirror beforehand and pheromone value could be shared on centralized resource management in server machine.

ACKNOWLEDGEMENTS

All authors are grateful for Universiti Teknologi MARA, Shah Alam, Selangor for financial support.

REFERENCES

[1] R. Rekaya et al. "Ant Colony Algorithm with Applications in the Field of Genomics." *Ant Colony Optimization-Techniques And Applications*. 2013.

[2] D.B. Mishra, A.A. Acharya, and R. Mishra. "Evolutionary algorithms for path coverage test data generation and optimization : a review." *Indonesian Journal of Electrical Engineering and Computer Science* Vol 15 No 1 pp 504-510, July 2019.

[3] H.N.K Al-Behadili, K. Ku-Mahamud and R. Sagban. "Annealing strategy for an enhance rule pruning technique in ACO-based rule classification." *Indonesian Journal of Electrical Engineering and Computer Science* Vol 16 No 3 pp 1499-1507, Dec 2019.

[4] M.A. Basir, Y. Yusof and M.S. Hussin. "Optimization of attribute selection model using bio-inspired algorithms." *Journal of ICT* 18 no.1 pp35-55, Jan 2019.

[5] M. Dorigo et al. "The ant system: An autocatalytic optimizing process." No. 91-016. Technical report, 1991.

[6] S.Bharath et al. "*Short text classification in twitter to improve information filtering*." In Proceedings of the *33rd international ACM* SIGIR conference on Research and development in information retrieval, pp. 841-842, 2010.

[7] F.Ronen, and J.Sanger. “The text mining handbook: advanced approaches in analyzing unstructured data.” *Cambridge University Press*, 2007.

[8] S. Goss et al. “Self-organized shortcuts in the Argentine ant.” *Naturwissenschaften*, 76(12), 579-581, 1989.

[9] M. Dorigo, and B. Mauro. "Ant colony optimization." In *Encyclopedia of machine learning*, pp. 36-39. Springer US, 2010.

[10] Z.A. Aziz. “*Ant colony hyper-heuristics for travelling salesman problem*.” Procedia Computer Science, 76, pp.534-538, 2015.

[11] A.A. Zalilah. “The Effect of Pheromone in Ant-Based Hyper-Heuristic.” In *Applied Mechanics and Materials* Vol. 446, pp. 1202-1206, 2014.

[12] M. Dorigo. "Optimization, learning and natural algorithms." Ph. D. Thesis, Politecnico di Milano, Italy (1992).

[13] M. Dorigo et al. "Ant system: optimization by a colony of cooperating agents." Systems, Man, and Cybernetics, Part B: *Cybernetics, IEEE Transactions* on 26, no. 1, pp29-41, 1996.

[14] D.L. Applegate et al. “The Traveling Salesman Problem: A Computational Study: A Computational Study.” *Princeton university press*, 2011.

[15] E.L. Lawler et al. "The traveling salesman problem. A guided tour of combinatorial optimisation.", 295, 1985.

[16] R. Gerhard. “The traveling salesman: computational solutions for TSP applications.” *Springer-Verlag*, 1994.

[17] M. Dorigo, and T. Stützle. "Ant colony optimization: overview and recent advances." In Handbook of metaheuristics, *Springer US*, pp 227-263, 2010.

[18] J.Tian, W.Yu, and S.Xie. "An ant colony optimization algorithm for image edge detection." In Evolutionary Computation, CEC 2008. *IEEE World Congress on Computational Intelligence*, pp. 751-756, 2008.

[19] M. Dorigo, and L. M. Gambardella. "Ant colonies for the travelling salesman problem." BioSystems 43, no. 2, pp73-81, 1997.

[20] H. Nezamabadi-pour, S. Saryazdi, and E. Rashedi. "Edge detection using ant algorithms." Soft Computing 10, no. 7, pp623-628, 2006.

[21] F.E.B. Otero, A.A. Freitas, C.G. Johnson. "cAnt-Miner: an ant colony classification algorithm to cope with continuous attributes." In Ant colony optimization and swarm intelligence. *Springer Berlin Heidelberg*, pp48-59, 2008.

[22] M. Dorigo et al. "Ant colony optimization." *Computational Intelligence Magazine, IEEE 1*, no. 4, 28-39, 2006.

[23] A.A. Freitas, H.S. Lopes and R.S. Parpinelli. "Ant colony algorithms for data classification." *Encyclopedia of Information Science and Technology* 1 pp154-159, 2008.

[24] R.W. Habtom et al. "Peak selection from MALDI-TOF mass spectra using ant colony optimization." *Bioinformatics 23*, no. 5, pp619-626, 2007

[25] B. Baharum, L.H. Lee, and K. Khan. "A review of machine learning algorithms for text-documents classification." *Journal of advances in information technology* 1, no. 1 (2010): 4-20.

[26] A. Pietramala et al. "A genetic algorithm for text classification rule induction." *In Machine Learning and Knowledge Discovery in Databases*, pp. 188-203. Springer Berlin Heidelberg, 2008.

[27] K. Emin, and O.K. Sahingoz. "*ACO algorithms with multi-core implementation*." Application of Information and Communication Technologies (AICT), 2013 7th International Conference on. IEEE, 2013.

[28] C.M. José, et al. "Enhancing data parallelism for ant colony optimization on GPUs." *Journal of Parallel and Distributed Computing* 73.1 (2013): 42-51.

[29] A. DeléVacq et al. "Parallel ant colony optimization on graphics processing units." *Journal of Parallel and Distributed Computing* 73.1 (2013): 52-61.

[30] M. Sharma, H. Elmiligi, and F. Gebali. "*SMARTs: A Tool to Simulate and Analyze the Performance of Real-time Multi-core Systems*." Procedia Computer Science 34 (2014): 544-551.

[31] P. Korošec et al. "Multi-core implementation of the differential ant-stigmergy algorithm for numerical optimization." *The Journal of Supercomputing* 63.3 (2013): 757-772.

[32] S.M Khalid et al. "Utilizing multiple pheromones in an ant-based algorithm for continuous-attribute classification rule discovery." *Applied Soft Computing* 13.1 (2013): 667-675.

[33] D. Moth. "Threading/Concurrency vs Parallelism". Retrieved from http://www.danielmoth.com/Blog/threadingconcurrency-vs-parallelism.aspx, 2008

[34] E. Singer. " The Remarkable Self-Organization of Ants ". Retrieved from https://www.quantamagazine.org/20140409-the-remarkable-self-organization-of-ants, 2014

[35] B. Roberto et al. "Genomic comparison of the ants Camponotus floridanus and Harpegnathos saltator." *science* 329.5995, 1068-1071, 2010

[36] W.D Tristram. "How animals communicate via pheromones." *American Scientist* 103.2, 114, 2015.

[37] R.L.W. Francis. "Trail Pheromones: An Integrative View of Their Role in Social Insect Colony Organization." *Annu. Rev. Entomol* 60, 581-99, 2015

[38] Y. Chen, L. Chen, and L. Tu. "Parallel ant colony algorithm for mining classification rules." *GrC*. 2006.

[39] R. Omid, and K. Zamanifar. "*Parallel ant miner 2*." International Conference on Artificial Intelligence and Soft Computing. Springer Berlin Heidelberg, 2008.

[40] L. Chengyong, L. Li, and Y. Xiang. "Research of multi-path routing protocol based on parallel ant colony algorithm optimization in mobile ad hoc networks*." Information Technology: New Generations*, 2008. ITNG 2008. Fifth International Conference on. IEEE, 2008.

[41] Jrennie, 20 Newsgroups. Retrieved from http://qwone.com/~jason/20Newsgroups, 2008

BIOGRAPHIES OF AUTHORS

Ahmad Nazmi Fadzal is a lecturer at UiTM Shah Alam in Faculty of Computer Sciences and Mathematic, Shah Alam. His degree in Computer Sciences was obtained from Kuliyyah of Information and Communication Technology in International Islamic University (IIUM), Gombak, Selangor, Malaysia in 2012 then pursue his master by research at Universiti Teknologi MARA Shah Alam under the supervision of Associate Professor Dr. Mazidah Puteh. After graduating he became a part-time lecturer in UiTM Kuala Terengganu, Terengganu. He is also a member of Research Interest Group (RIG) MuDIR, which represents for Multi-Disciplinary Information Retrieval, under the community of research (CORE) Advanced Computing and Communication, UiTM.

Mazidah Puteh (PhD) is an Associate professor of Computer Science Department at Faculty of Computer and Mathematical Sciences, Universiti Teknologi MARA, Terengganu Branch, Malaysia. She has 28 years of teaching experiences in the field of Computer Science. She also has more than 10 years of research and publications in the area of machine learning especially in bio-inspired computational intelligence. Her area of interest are data science, data mining and optimization.

Nurazzah Abdul Rahman (PhD) is an Associate Professor of Computer Science Department, Faculty of Computer and Mathematical Sciences, Universiti Teknologi MARA, Shah Alam, Malaysia. Her main research area is Information Retrieval (IR), focuses in Malay Text IR specifically developing and manipulating Malay Translated Hadith Text corpus and Information Extraction. She is also an active member of IEEE CS Malaysia Chapter and the Society of Information Retrieval and Knowledge Management (PECAMP). Together with other Executive Members, she has been organizing IEEE flagship conferences, workshops and Distinguished Visitor Program (DVP) for IEEE members in Malaysia from 2014 to 2017, also PECAMP flagship conferences: CAMP’10, CAMP’12,CAMP’16 and CAMP’18. Currently she is the Head of Research Interest Group (RIG) MuDIR registered under Universiti Teknologi MARA.