ISSN: 0067-2904

Distributed Multi-Ant Colony System Algorithm using Raspberry Pi Cluster for Travelling Salesman Problem

Mustafa Muwafak Alobaedy[1]*, Ali A. Khalaf[2], Yousef Fazea[3]
[1]*School of Information & Communication Technology, HELP University, Kuala Lumpur, Malaysia*
[2]*Computer Science Department, College of Science, University of Baghdad, Baghdad, Iraq*
[3]*Department of Computer & Information Technology, Marshall University, 1 John Marshall Drive, Huntington, WV 25755, USA*

Received: 25/10/2021 Accepted: 27/4/2022 Published: 30/9/2022

Abstract

The traveling salesman problem is addressed in this paper by introducing a distributed multi-ant colony algorithm that is implemented on a Raspberry Pi cluster. The implementation of a master and eight workers, each running on Raspberry Pi nodes, is the central component of this novel technique. Each worker is responsible for managing their own colony of ants, while the master coordinates communications among workers' nodes and assesses the most optimal approach. To put the newly built cluster through its paces, several datasets of traveling salesman problem are used to test the created cluster. The findings of the experiment indicate that a single board computer cluster, which makes use of multi-ant colony algorithm, is a viable alternative for distributed computing. This approach's extensibility options are extensively discussed as well.

Keywords: Metaheuristic Algorithm, Single-Board Computing Cluster, Combinatorial Optimization problem, Distributed Computing, Distributed Algorithm.

1. Introduction

The Ant Colony System (ACS) algorithm has been successfully applied in a wide range of combinatorial optimization problems [1]. The mechanism of the ACS algorithm is inspired by the natural behavior of biological ant colonies [2][3]. The ACS algorithm is built on notation of the reinforcement learning concept [4]. It uses ants as agents to manipulate environment via use of pheromone trails to find the shortest path. In every iteration cycle, each ant builds up a new solution step by step using exploitation and exploration mechanisms. Simultaneously, each ant reduces the pheromone from the path it uses to dissuade the next ant from utilizing the same path, causing the following ants to explore other paths (solutions). The determined as the best path so far among all the ants is labeled by applying pheromone to the edges of that path. In the subsequent iteration, the ants are drawn to search in the solution space near the previous best path in the succeeding iteration [5], [6]. The ACS algorithm may be implemented in a number of ways, including single or multi colonies [2].

Despite the fact that the ACS algorithm has been successfully implemented sequentially in a variety of domains, little attention has been paid to distributed multi-ant colony implementation [7]. Therefore, ACS algorithm could be a promising candidate for parallelization.

*Email: new.technology@hotmail.com

In this paper, the Multi Ant Colony System (MACS) which comprises of Raspberry Pi cluster with nodes (workers), is used to investigate distributed multi ant colony based on coarse-grained approach. Each node represents an entire ant colony and a master node has been deployed to coordinate the workers activities. MACS is an iterative technique in which the colonies transfer their solution to the master node for evaluation after each iteration. Using the 3-opt approach, the master node will choose the optimal option for further improvement. If the improved solution outperforms the previous best so far solution, then it will be preserved. If the stop condition is not met, the master node will apply a global pheromone update; if it is met, the algorithm will terminate the iteration and display the final solution. Consequently, the created cluster's performance was tested and evaluated using dataset token from traveling salesmen problem. The findings show that the Raspberry Pi cluster achieved a significant performance across three datasets. Thus, the proposed cluster could be scaled to a wide range of combinatorial optimization problems and could be expanded to any number of workers. Therefore, the proposed cluster is an important contribution to the body of knowledge.

The following sections outline the structure of this paper. Section 2 presents an overview on the Raspberry Pi. Section 3 presents the classifications of Raspberry Pi cluster applications based on their functionalities. Section 4 discusses the methodology used. Then the findings are discussed in Section 5. Section 6 presents the conclusion and future directions.

2. Raspberry Pi Overview

The Raspberry Pi (RPi) was initially developed by the Raspberry Pi Foundation to encourage the teaching of computer science and basic electronic courses in developing countries [8][9]. RPi gained popularity after researchers and practitioners found that it is a useful and practical device for a variety of applications including robotics, weather monitoring, and smart homes [10]–[12].

The RPi is depicted in Figure 1. It is a single-board computer with dimensions of approximately 8.8 x 5.8 x 1.8 cm. The RPi can run a Linux-based operating system (OS), known as Raspberry Pi OS, which comes with three options: Lite; desktop; and desktop with recommended software. In addition, the RPi can also support third-party OS as well, e.g., Raspbian, RISC OS, Windows IOT Core, and etc.

Figure 1-Raspberry Pi 4 Model B dimensions

The RPi comes in a variety of hardware models with varying memory, networks, and processing capabilities. Table 1 summarizes some of the features of several RPi models [13][14]. At the time of writing, the latest model of the RPi, released in 2019, is the Raspberry Pi 4 Model B which offers a processing speed of 1.5 GHz (quad-core), up to 8 GB memory,

SD card support, GPIO, Wi-Fi, LAN Gigabit Ethernet, 4 USB, and two micro-HDMI ports (support up to 4 Kp 60). The RPi is therefore a useful and practical choice for a low-cost computing cluster capable of successfully solving small and medium-scale problems.

Table 1- Comparison between Raspberry Pi models

Model	Release Date	CPU	Memory	Network
Raspberry Pi 4 B	2019	Quad-core (1.5GHz)	2, 4, 8 GB	Gigabit Ethernet, Wireless and Bluetooth
Raspberry Pi 3 A+	2018	Quad-core (1.4GHz)	512MB	Wireless and Bluetooth
Raspberry Pi 3 B+	2018	Quad-core (1.4GHz)	1GB	Gigabit Ethernet, Wireless and Bluetooth
Raspberry Pi 3 B	2016	Quad-core (1.2GHz)	1GB	Wireless LAN / 100 Base Ethernet
Raspberry Pi 2 B	2015	Quad-core (900MHz)	1GB	100 Base Ethernet
Raspberry Pi 1 Model A+	2014	Single core (700MHz)	512MB	No
Raspberry Pi 1 Model B+	2014	Single core (700MHz)	512MB	100 Base Ethernet
Raspberry Pi Zero W	2017	Single core (1GHz)	512MB	Wireless LAN / Bluetooth
Raspberry Pi Zero	2015	Single core (1GHz)	512MB	No

3. Classification of Raspberry Pi Cluster Applications

Figure 2 depicts the graphical representation of the taxonomy categorization of RPi application-oriented based. The taxonomy classification of RPi cluster applications was developed by conducting an extensive literature analysis of the existing literature.

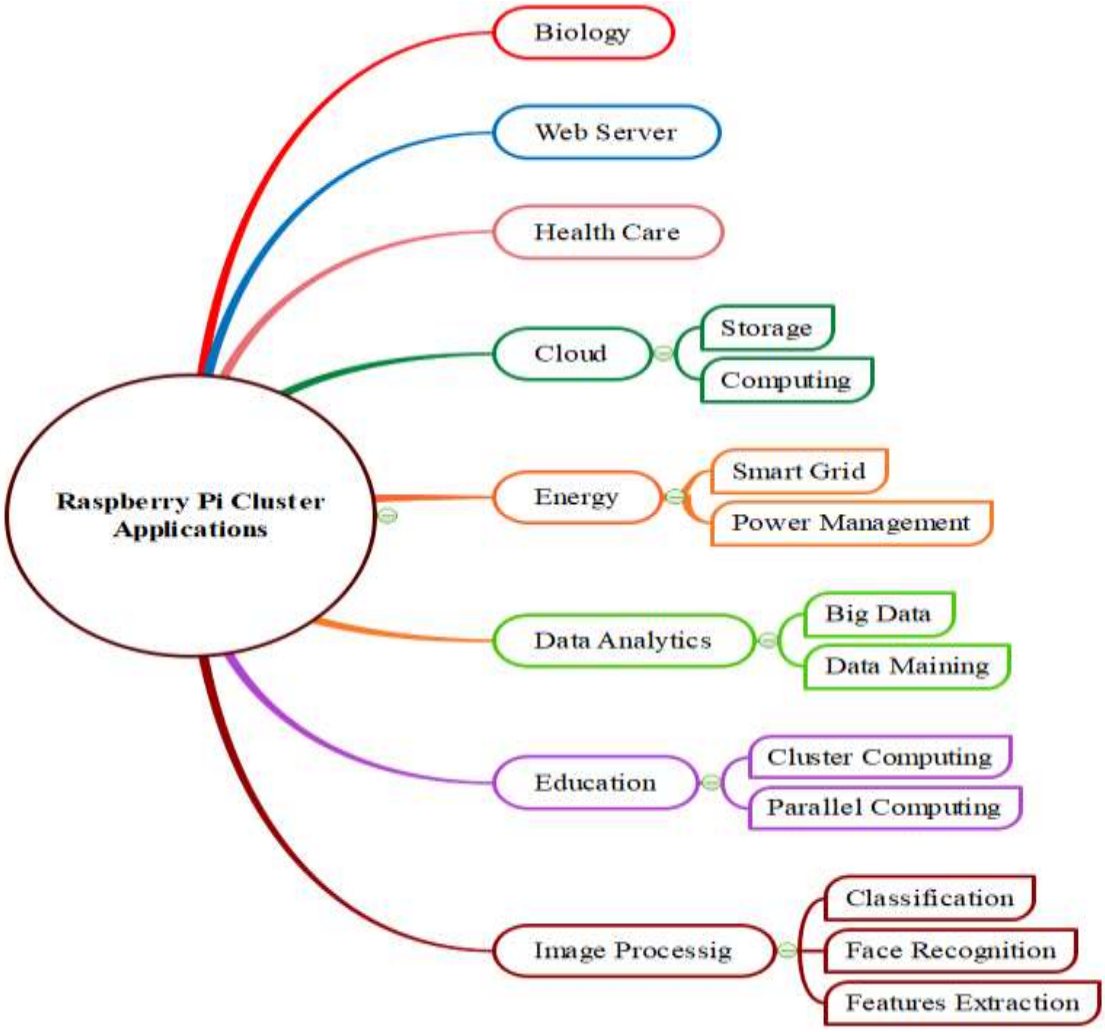

Figure 2-Mind-map of domains identified in Raspberry Pi cluster applications

Data Analytics

Data analytics requires high-performance computing in a data center to process intensive computing tasks such as classification and visualization. A viable alternative to an expensive data center is a Hadoop cluster using single-board computing such as the RPi. Despite its low-power processors, it is possible to use RPi in big data analytics by combining it with several RP is in a cluster [15]. RPi cluster technology is applied in various types of data analytics applications such as big data, data mining, and visualization. Open-source software such as Message Passing Interface (MPI), Hadoop, Spark, and Yet Another Resource Negotiator (YARN) can be run from RPi cluster [16]–[19].

RPi cluster was implemented to process tourists' data collected from Facebook, Twitter, Foursquare, and Instagram APIs to generate a heatmap [17]. The heatmap displays the tourists' geolocations, comments, and attractions. A RPi cluster has similar capabilities as fully functional servers for big data and video streaming applications in centers [18]. In addition, RPi cluster is used in data mining to solve classification problems using the K-Means algorithm in big data [16]. RPi clusters are expected to play a prominent role in data analytics for small organizations.

Education

The idea behind creating the RPi device was to encourage students to learn programming and to develop their interest in computer science concepts, especially for students in developing countries [8], [9]. In higher education, teaching subjects such as High-Performance Computing (HPC), cluster and distributed computing with hands-on experience is very limited due to the high cost of the equipment [20]. Since the low cost of RPi makes it a viable option to be incorporated into the curriculum, RPi cluster has since been introduced in practical HPC education activities [19]–[21]. RPi clusters are also used as dedicated parallel computing hardware for each student, instead of students having to share hardware [22]. With the benefits of remote access, RPi cluster enables students to build their own File Systems in User Space (FUSE), and gain experience of using a low-cost remote laboratory [23], [24]. In addition, RPi cluster can potentially be extended into a mini supercomputer [25], [26], that can be used for education and research as a high-performance computer. The advantages of low cost, mobility, size, weight, and ambient cooling might change the way parallel and distributed computing is taught.

Image Processing

RPi cluster with open-source libraries such as Hadoop and OpenCV could play a pivotal role in image processing e.g. feature points extraction system using Speeded Up Robust Features (SURF) algorithm [27], face recognition [28], image stacking [29], edge detection [30], [31], image analysis [32], ray tracing [33], image conversion [34], image recognition [35], Fourier transform [36] and image classification [37]. Moreover, a RPi cluster can extend its capability in image processing by adding a GPU [38].

Cloud Technology

RPi cluster has been implemented in two types of cloud technology: computing and storage. RPi cloud computing was developed using Nextcloud, an open-source client-server software [39]. To overcome the lack of computing power of the RPi, a cloud computing environment was developed using 300 nodes; this effectively demonstrated that RPi could be an inexpensive and green alternative for cloud computing in research [40]. RPi cloud has the potential to provide various cloud layers such as cloud stack, resource virtualization, and network for the educational environment [41]. Furthermore, the RPi hardware and the crowdsourcing platform could support the development of crowd cloud: a crowdsourced system for cloud infrastructure, cloud platform, and cloud software services [42]. A naïve model of HPC as a service was implemented using the RPi cluster in [43]. Moreover, RPi cluster has the capability to be used as cloud storage to provide real-time applications [44].

Miscellaneous Applications

We found several additional applications of RPi clusters in other domains such as anomaly detections in the smart grid [45], power management [46], genome pattern matching in biological computation [47], patients monitoring systems in health care [48], and load balancing methods in webservers [49].

4. Methodology for Distributed Multi-Ant Colony System Algorithm

In this investigation, we employed a Raspberry Pi cluster with one master and eight worker nodes. As indicated in Figure 3, the cluster is linked via a switch that supports one-gigabit connection.

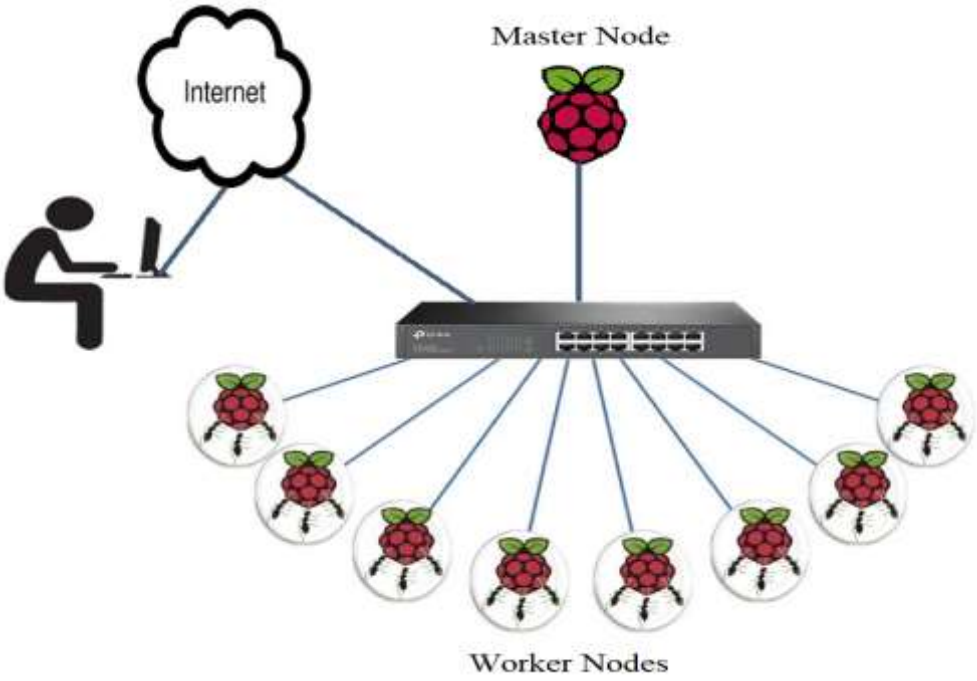

Figure 3- Raspberry Pi cluster network topography

The application was developed with the C#.NET framework and the Mono open-source library. The experiments were carried out utilizing eleven symmetric Travelling Salesman Problem (TSP) datasets, each of which reflects a different size and complexity of the work environment. Undermaster node coordination, each worker was implemented as a complete ant colony using the Ant Colony System (ACS) algorithm. Each ant colony node was populated with eight ants, which were chosen by following a pilot experiment to determine the optimal number of ants. The master node begins by identifying the local network's linked workers, then begins reading the TSP dataset for the initialization phase, which includes calculating distance, creating initial pheromone, and generating the initial solution. The master node begins delivering tasks and basic information to the workers at this stage.

Using the pheromone matrix obtained from the master, each worker begins looking for the optimum solution and applying local pheromone updates. The worker will communicate the best solution to the master node once it achieves the stop condition. The master node will assess the solutions received from the worker nodes and choose the best one. The master node used the 3-Opt local search method to get the best solution. If the best employee's solution is better than the previous solution, it will be saved as the best so far solution. If the termination condition is fulfilled, the master node applies a global pheromone update, and the entire process is replayed using the updated pheromone matrix. The procedure will be ended if the termination condition is satisfied, and the best solution so far will be shown. The process flow for master and worker nodes is depicted in Figure 4.

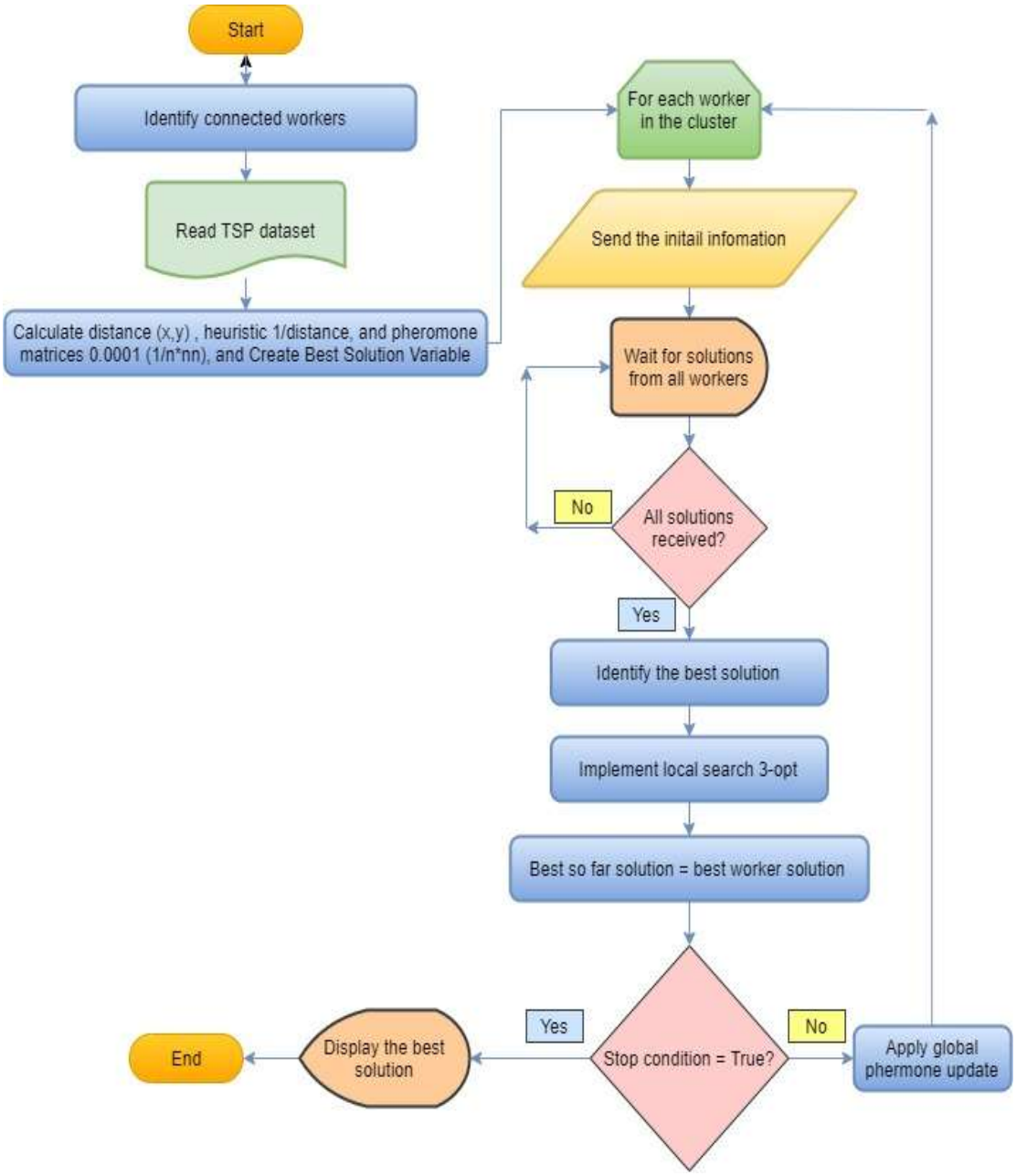

Figure 4-Master and workers process flow for Raspberry Pi cluster.

Developing ACS for Raspberry Pi Cluster

This study implemented Raspberry Pi nodes as a computing cluster that consists of a master node and eight worker nodes. The procedure begins with the master node connecting the workers through IP address as shown in Figure 5. The next phase will be carried out once the connection has been established.

- Read TSP instances
- Initialize distance (d_{ij}) using Euclidian distance, which generates a symmetric initial distance matrix $(d_{ij} = (d_{ji}))$.
- Initialize heuristic $(h_{ij}) = 1/d_{ij})$, where d_{ij} is the distance from city i to city j.
- Initialize the pheromone matrix (ph_{ij}) using the nearest neighbor approach $(\tau_0 = \frac{1}{C^{nn}})$.
- Initialize a place holder for best so far solution $\hat{S}$.
- Master sends the initial information to each worker.

procedure ACS master
$w \leftarrow$ List of Workers IP Address
Read TSP Dataset
$d \leftarrow$ Initial Distance
$h \leftarrow$ Initial Heuristic
$ph \leftarrow$ Initial Pheromone
$\hat{S} \leftarrow$ Initial best so far, a solution
While termination condition is not met:
$\boldsymbol{For\ i = 1\ to\ k\ Do}$:
Send Initial Information to w_i $(d, h, ph, \hat{S})$
w_i starts ACS algorithm (Figure No)
$\boldsymbol{End - for}$
While not all solution received from w do:
Wait for Solution
End-While
$S_{bi} \leftarrow$ Select the Best Iteration Solution
$S_{3opt} \leftarrow$ Implement 3-opt (h, S_{bi})
if $(f(S_{3opt}) < f(\hat{S})$ then
$\hat{S} \leftarrow S_{3opt}$
$ph \leftarrow best\ ant\ pheromone$
end-if
Apply Global Update Pheromone
End-While
Display $\hat{S}$
End-procedure

Figure 5-Master implementation for TSP using Raspberry Pi nodes based on Distributed ACS algorithm.

The master node's primary responsibility is to set up the relevant parameters, such as distance, heuristic, and pheromone, and then transfer that information to the worker nodes for processing. Worker nodes are in charge of determining the optimal solution. Using the pseudorandom proportional rule [8], each worker implements eight ants to find the optimal solution. In addition, as illustrated in Figure 6, employees perform local pheromone updates throughout solution construction and global pheromone updates to the best so far solution in the colony.

Procedure ACS-Worker
Receive Initial Information to w_i $(d, h, ph, \hat{S})$
Initialize the number of ants n
While (Termination condition not met) Do
For i $= 1$ to n Do:
Construct new solution
Apply local pheromone update
End For
Apply Global pheromone update
Update best found solution s^*
End-While
End-procedure

Figure 6-Worker's implementation for TSP using Raspberry Pi nodes based on Distributed ACS algorithm.

Experiment Design and Datasets

The experiment was conducted using symmetric TSP datasets obtained from the TSPLIB library [9]. Each dataset contains a different number of cities and some of them have been provided with optimum paths. TSP is a problem of finding the shortest tour starting from any city, visiting every city one time only and going back to the starting city. Formally, TSP is a fully connected graph $G = (N, A)$, where N represents the cities and A represents the connections. Each connection $(i, j) \in A$ is assigned with distance (d) value between (i, j). In this study, we have implemented an asymmetric TSP where $d_{ij} = d_{ji}$. The problem can be formulated mathematically as shown in equation 1:

$$f(\pi) = \sum_{i-1}^{n-1} d_{\pi(i)\pi(i+1)} + d_{\pi(n)\pi(1)}. \quad (1)$$

Finding the shortest path in TSP is classified as an $NP - hard$ optimization problem that requires a metaheuristic algorithm such as Genetic Algorithm (GA), Simulated Annealing (SA), and Ant Colony Optimization [10].

We have selected eleven TSP datasets for experiments, each experiment conducted 30 times to obtain the best, worse, mean, and standard deviation. Some datasets have optimal solutions provided from the source while others do not have.

ACS Parameters

Table 2 provides the parameters for the implemented ACS.

Table 2-ACS Parameters

Parameters for All Datasets	
No of Ant Colony	8
No of Ants in each Colony	8
Initial Pheromone	1 / (8 * nn *(Nearest-neighbor))*
No of Iterations	20 *(Main ACS)* x 100 *(Sub-ACS)*
Alpha (α)	1
Beta (β)	2
Raw (ρ)	0.1
Zeta (ξ)	0.1
q0	0.9

5. Results and Discussion

Various studies conducted experiments using TSP datasets indicate that metaheuristic algorithms have different performances on the deferent dataset. [5] presented the results of TSP using ant colony optimization algorithms such as Ant System (AS), Max-Min Ant System (MMAS), and Ant Colony System (ACS). In our study, we have implemented the ACS algorithm in the Raspberry Pi cluster based on distributed ant colonies. We have obtained the optimum traveling distance for three instances. Table 3 and Figure 7 present the experiments results (consider this instead- results of the experiment.)

Table 3-Results of DMACO using eleven TSP datasets

No	Dataset	Best	Worst	Mean	SD	Optimum
1	eil51	**426**	437	431.17	2.28	**426**
2	att48	**33522**	34063	33772.00	161.75	**33522**
3	eil76	**538**	552	547.37	2.81	**538**
4	st70	681	692	687.07	3.36	675
5	d198	15922	16100	16025.80	43.14	*
6	eil101	640	652	644.97	3.47	629
7	kroA100	21512	22124	21814.50	160.87	21282
8	rat99	1223	1259	1243.17	8.35	*
9	rat195	2047	2085	2068.63	9.40	*
10	rat783	10853	10950	10903.36	28.49	*
11	pcb442	52645	53321	53034.93	183.15	50778

*Not provided in the original dataset.

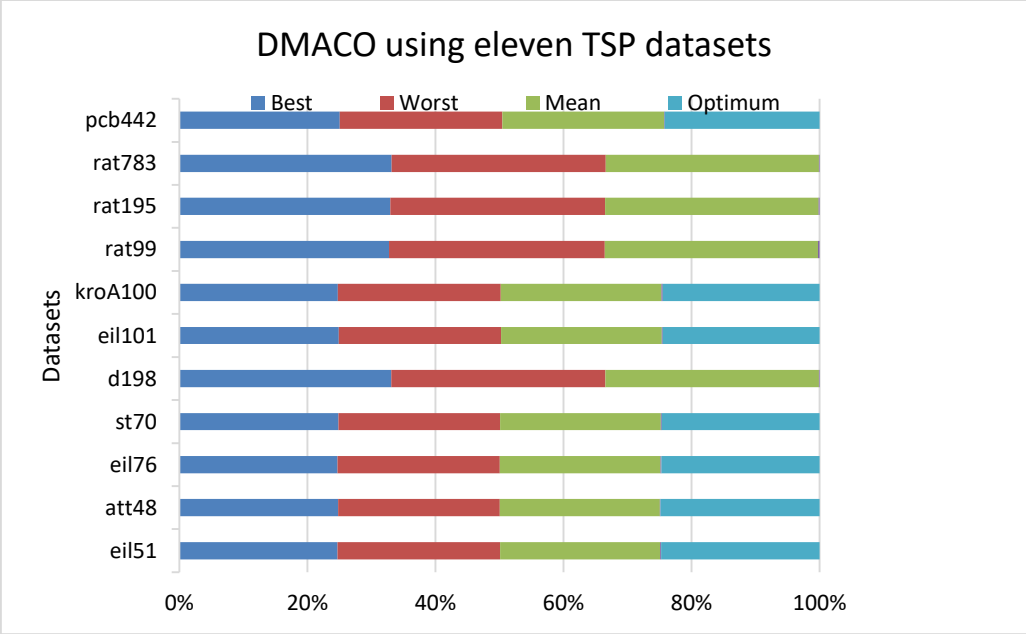

Figure 7- DMACO using eleven TSP datasets

The findings reveal that in three datasets, namely eil51, att48, and eil76, the Raspberry Pi cluster was able to reach the best answer. Despite the small number of optimal outcomes found in this study, the standard deviation values demonstrate that the worst values are near to the mean values, indicating that cluster performance is consistent across most datasets. The findings show that a single board computing device like the Raspberry Pi is a good alternative for a low-cost, small-scale cluster to handle optimization issues like TSP.

6. Limitations

RPi cluster demonstrate very good performance in small and medium scale datasets. However, RPi cluster is not practical for large scale datasets due to the long processing time required to solve such problems.

7. Conclusion

In this experiment, we have proved that a single board computer may be used as a processing unit in a computing cluster. Therefore, TSP datasets and the ACS technique were used to test the constructed cluster. We also demonstrated that each Raspberry Pi may be configured as a self-contained worker with master coordination. Experiments indicates that the Raspberry Pi cluster is appropriate for small and medium-sized challenges. We have also included the

network topology, process flow, and implementation pseudocode. The Raspberry Pi cluster might be used for a variety of combinatorial optimization problems, allowing for future study and development.

Acknowledgment

The authors would like to thank the Computer Science Department, College of Science, University of Baghdad.

References

[1] B. Chandra Mohan and R. Baskaran, "A survey: Ant Colony Optimization based recent research and implementation on several engineering domain," *Expert Syst. Appl.*, vol. 39, no. 4, pp. 4618–4627, 2012, doi: https://doi.org/10.1016/j.eswa.2011.09.076.

[2] M. Dorigo and T. Stützle, *Ant colony optimization*. Cambridge, Mass: MIT Press, 2004.

[3] S. Fidanova, "Ant Colony Optimization," in *Studies in Computational Intelligence*, vol. 947, Cambridge, Mass: MIT Press, 2021, pp. 3–8.

[4] S. Bromuri, "Dynamic heuristic acceleration of linearly approximated SARSA: using ant colony optimization to learn heuristics dynamically," *J. Heuristics*, vol. 25, no. 6, pp. 901–932, Dec. 2019, doi: 10.1007/s10732-019-09408-x.

[5] M. Middendorf, F. Reischle, and H. Schmeck, "Multi Colony Ant Algorithms," *J. Heuristics*, vol. 8, no. 3, 2002, doi: doi.org/10.1023/A:1015057701750.

[6] D. Kaeli, P. Mistry, D. Schaa, and D. P. Zhang, *Heterogeneous Computing with OpenCL 2.0*. Waltham: Elsevier, 2015.

[7] M. Starzec, G. Starzec, A. Byrski, and W. Turek, "Distributed ant colony optimization based on actor model," *Parallel Comput.*, vol. 90, p. 102573, Dec. 2019, doi: 10.1016/j.parco.2019.102573.

[8] N. S. Yamanoor and S. Yamanoor, "High Quality, Low Cost Education with the Raspberry Pi," in *Proceedings of IEEE Global Humanitarian Technology Conference (GHTC)*, 2017, pp. 1–9, doi: 10.1109/GHTC.2017.8239274.

[9] N. Valov and I. Valova, "Raspberry Pi as a Tool to Combine Different Courses Part of University Education," in *Proceedings of the 18th International Conference on Information Technology Based Higher Education and Training (ITHET)*, 2019, pp. 1–5, doi: 10.1109/ITHET46829.2019.8937334.

[10] J. Cicolani, *Beginning Robotics with Raspberry Pi and Arduino : using Python and OpenCV*. New York: Apress, 2018.

[11] A. J. Lewis, M. Campbell, and P. Stavroulakis, "Performance evaluation of a cheap, open source, digital environmental monitor based on the Raspberry Pi," *Measurement*, vol. 87, pp. 228–235, 2016, doi: 10.1016/j.measurement.2016.03.023.

[12] S. Goodwin, *Smart Home Automation with Linux and Raspberry Pi*. Berkeley, CA New York: Apress, 2013.

[13] P. Fromaget, *Master your Raspberry Pi in 30 Days*. Independently published, 2020.

[14] Raspberrypi.org, "Raspberry Pi Products," 2020. https://www.raspberrypi.org/products/.

[15] C. Kaewkasi and W. Srisuruk, "A study of big data processing constraints on a low-power hadoop cluster," in *Proceedings of the International Computer Science and Engineering Conference, ICSEC*, 2014, pp. 267–272, doi: 10.1109/ICSEC.2014.6978206.

[16] J. Saffran *et al.*, "A Low-Cost Energy-Efficient Raspberry Pi Cluster for Data Mining Algorithms," in *Euro-Par 2016: Parallel Processing Workshops. Euro-Par 2016. Lecture Notes in Computer Science*, F. Desprez, P.-F. Dutot, C. Kaklamanis, L. Marchal, K. Molitorisz, L. Ricci, V. Scarano, M. A. Vega-Rodríguez, A. L. Varbanescu, S. Hunold, S. L. Scott, S. Lankes, and J. Weidendorfer, Eds. Springer, Cham, 2016, pp. 788–799.

[17] M. D'Amorea, R. Baggiob, and E. Valdania, "A practical approach to big data in tourism: a low cost raspberry pi cluster," in *Proceeding of 22nd International Conference on Information Technology and Travel & Tourism*, 2015, pp. 169–181, doi: 10.1007/978-3-319-14343-9_13.

[18] N. J. Schot, P. J. E. Velthuis, and B. F. Postema, "Capabilities of Raspberry Pi 2 for Big Data and Video Streaming Applications in Data Centres," in *Measurement, Modelling and Evaluation of Dependable Computer and Communication Systems*, A. Remke and B. R. Haverkort, Eds.

Springer, Cham, 2016, pp. 183–198.

[19] S. J. Cox, J. T. Cox, R. P. Boardman, S. J. Johnston, M. Scott, and N. S. O'Brien, "Iridis-pi: a low-cost, compact demonstration cluster," *Cluster Comput.*, vol. 17, no. 2, pp. pages349–358, 2013, doi: 10.1007/s10586-013-0282-7.

[20] A. M. Pfalzgraf and J. A. Driscoll, "A low-cost computer cluster for high-performance computing education," in *Proceeding of IEEE International Conference on Electro Information Technology*, 2014, pp. 362–366, doi: 10.1109/EIT.2014.6871791.

[21] S. Mollova, M. Zhekov, A. Kostadinov, and P. Georgieva, "Laboratory model for research on computer cluster systems," in *Proceeding of the 41st International Convention on Information and Communication Technology, Electronics and Microelectronics (MIPRO)*, 2018, pp. 1388–1393, doi: 10.23919/MIPRO.2018.8400250.

[22] B. Y. Shen and S. Mukai, "A Portable, Inexpensive, Nonmydriatic Fundus Camera Based on the Raspberry Pi®Computer," *J. Ophthalmol.*, 2017, doi: 10.1155/2017/4526243.

[23] W. J. Keeler and J. Wolfer, "A Raspberry PI cluster and Geiger counter supporting random number acquisition in the CS Operating Systems class," in *Proceeding of 13th International Conference on Remote Engineering and Virtual Instrumentation (REV)*, Feb. 2016, pp. 353–354, doi: 10.1109/REV.2016.7444500.

[24] S. Mollova, K. Seymenliyski, S. Letskovska, R. Simionov, and E. Zaerov, "Training System for Studing Computer Clusters," in *Proceeding of the International Conference on High Technology for Sustainable Development (HiTech)*, 2019, pp. 8–11.

[25] T. K. Priyambodo, A. W. Lisan, and M. Riasetiawan, "Inexpensive Green Mini Supercomputer Based on Single Board Computer Cluster," *J. Telecommun. Electron. Comput. Eng.*, vol. 10, no. 1, pp. 141–145, 2018.

[26] P. Turton and T. F. Turton, "Pibrain - A cost-effective supercomputer for educational use," in *Proceeding og the 5th Brunei International Conference on Engineering and Technology (BICET)*, 2014, vol. 2014, pp. 2–5, doi: 10.1049/cp.2014.1121.

[27] K. Srinivasan, C. Y. Chang, C. H. Huang, M. H. Chang, A. Sharma, and A. Ankur, "An Efficient Implementation of Mobile Raspberry Pi Hadoop Clusters for Robust and Augmented Computing Performance," *J. Inf. Process. Syst.*, vol. 14, no. 4, pp. 989–1009, 2018, doi: 10.3745/JIPS.01.0031.

[28] S. R. Rudraraju, N. K. Suryadevara, and A. Negi, "Face Recognition in the Fog Cluster Computing," in *Proceeding of IEEE International Conference on Signal Processing, Information, Communication and Systems, SPICSCON*, 2019, pp. 45–48, doi: 10.1109/SPICSCON48833.2019.9065100.

[29] M. G. Perna *et al.*, "First approach to image stacking using a Single-Board Computer - A small study of strengths, opportunities, weaknesses and threats," in *Proceeding of Congreso Argentino de Ciencias de la Informatica y Desarrollos de Investigacion, CACIDI*, 2018, vol. 53, pp. 1–5, doi: 10.1109/CACIDI.2018.8584353.

[30] D. Marković, D. Vujičić, D. Mitrović, and S. Ranđić, "Image Processing on Raspberry Pi Cluster," *Int. J. Electr. Eng. Comput.*, vol. 2, no. 2, 2018, doi: 10.7251/IJEEC1802083M.

[31] V. Govindaraj, "Parallel Programming in Raspberry Pi Cluster," 2016.

[32] B. Qureshi, Y. Javed, A. Koubâa, M. F. Sriti, and M. Alajlan, "Performance of a Low Cost Hadoop Cluster for Image Analysis in Cloud Robotics Environment," *Procedia Comput. Sci.*, vol. 82, pp. 90–98, 2016, doi: 10.1016/j.procs.2016.04.013.

[33] C. Baun, "Parallel image computation in clusters with task-distributor," *Springerplus*, vol. 6, no. 632, 2016, doi: 10.1186/s40064-016-2254-x.

[34] R. F. Rahmat, T. Saputra, A. Hizriadi, T. Z. Lini, and M. K. M. Nasution, "Performance Test of Parallel Image Processing Using Open MPI on Raspberry PI Cluster Board," in *Proceeding of 3rd International Conference on Electrical, Telecommunication and Computer Engineering, ELTICOM*, 2019, pp. 32–35, doi: 10.1109/ELTICOM47379.2019.8943848.

[35] T. Rausch, C. Avasalcai, and S. Dustdar, "Portable Energy-Aware Cluster-Based Edge Computers," in *Proceedings of IEEE/ACM Symposium on Edge Computing (SEC)*, 2018, pp. 260–272, doi: 10.1109/SEC.2018.00026.

[36] D. Vujičić, D. Mitrović, D. Marković, M. Vesković, and S. Ranđić, "Practical Aspects of Using Virtualization with Raspberry Pi Clusters," in *Proccdeding of UNITECH International Scientific*

Conference, 2018, pp. 113–116, [Online]. Available: https://www.researchgate.net/publication/329045116.

[37] Z. Huang, "Real-time Pedestrian Classification System Using Deep Learning on a Raspberry Pi Cluster," University of Calgary, 2019.

[38] J. An, S. Park, and I. Ihm, "Construction of a flexible and scalable 4D light field camera array using Raspberry Pi clusters," *Vis. Comput.*, vol. 35, no. 10, pp. 1475–1488, 2019, doi: 10.1007/s00371-018-1512-z.

[39] B. Sukesh, K. Venkatesh, and L. N. B. Srinivas, "A Custom Cluster Design With Raspberry Pi for Parallel Programming and Deployment of Private Cloud," in *Role of Edge Analytics in Sustainable Smart City Development*, G. R. Kanagachidambaresan, Ed. Hoboken, NJ: Wiley-Scrivener., 2020, pp. 273–288.

[40] P. Abrahamsson *et al.*, "Affordable and Energy-Efficient Cloud Computing Clusters: The Bolzano Raspberry Pi Cloud Cluster Experiment," in *Proceeding of IEEE 5th International Conference on Cloud Computing Technology and Science*, Dec. 2013, pp. 170–175, doi: 10.1109/CloudCom.2013.121.

[41] F. P. Tso, D. R. White, S. Jouet, J. Singer, and D. P. Pezaros, "The Glasgow Raspberry Pi Cloud: A Scale Model for Cloud Computing Infrastructures," in *Proceeding of IEEE 33rd International Conference on Distributed Computing Systems Workshops*, Jul. 2013, pp. 108–112, doi: 10.1109/ICDCSW.2013.25.

[42] M. Hosseini, C. M. Angelopoulos, W. K. Chai, and S. Kundig, "Crowdcloud: a crowdsourced system for cloud infrastructure," *Cluster Comput.*, vol. 22, no. 2, pp. 455–470, 2019, doi: 10.1007/s10586-018-2843-2.

[43] H. A. Imran, S. Wazir, A. J. Ikram, A. A. Ikram, H. Ullah, and M. Ehsan, "HPC as a Service: A naïve model," in *Proceeding of 8th International Conference on Information and Communication Technologies, ICICT*, 2019, pp. 174–179, doi: 10.1109/ICICT47744.2019.9001912.

[44] S. E. Princy and K. G. J. Nigel, "Implementation of cloud server for real time data storage using Raspberry Pi," in *Proceedings of Online International Conference on Green Engineering and Technologies (IC-GET)*, 2016, pp. 25–28, doi: 10.1109/GET.2015.7453790.

[45] K. Candelario, C. Booth, A. St Leger, and S. J. Matthews, "Investigating a Raspberry Pi cluster for detecting anomalies in the smart grid," in *Proceeding of IEEE MIT Undergraduate Research Technology Conference, URTC*, 2018, pp. 1–4, doi: 10.1109/URTC.2017.8284197.

[46] M. F. Cloutier, C. Paradis, and V. M. Weaver, "A Raspberry Pi Cluster Instrumented for Fine-Grained Power Measurement," *Electronics*, vol. 5, no. 4, 2016, doi: 10.3390/electronics5040061.

[47] P. Kanani and M. Padole, "Improving Pattern Matching performance in Genome sequences using Run Length Encoding in Distributed Raspberry Pi Clustering Environment," *Procedia Comput. Sci.*, vol. 171, pp. 1670–1679, 2020, doi: 10.1016/j.procs.2020.04.179.

[48] S. Misbahuddin, A. R. Al-Ahdal, and M. A. Malik, "Low-Cost MPI Cluster Based Distributed in-Ward Patients Monitoring System," in *Proceedings of IEEE/ACS International Conference on Computer Systems and Applications, AICCSA*, 2019, pp. 1–6, doi: 10.1109/AICCSA.2018.8612824.

[49] M. W. P. Maduranga and R. G. Ragel, "Comparison of load balancing methods for Raspberry-Pi Clustered Embedded Web Servers," in *Proceeding of International Computer Science and Engineering Conference (ICSEC)*, Dec. 2016, pp. 1–4, doi: 10.1109/ICSEC.2016.7859875.