IOSR Journal of Computer Engineering (IOSR-JCE)
e-ISSN: 2278-0661,p-ISSN: 2278-8727, Volume 19, Issue 4, Ver. IV. (Jul.-Aug. 2017), PP 80-87
www.iosrjournals.org

Optimising Task management of Robot and deciding whether Cloud Computing is feasible

*Amitava Kar[1], Ajoy K. Dutta[2], Subir K. Debnath[3]
[1](Department of Computer Application, Asansol Engineering College, India)
[2,3](Production Engineering Department, Jadavpur University, India)
*Corresponding Author: *Amitava Kar*

Abstract: *In this paper, we describe the application of cloud computing in the field of robotics where a robot is assigned the job of picking up jobs lying in the floor and to keep them in the given space. The robot is connected to the cloud through a network. It receives the answer to the queries of the database residing in the cloud about the location of the jobs to be picked up and receives the coordinates of the location of the spaces where it is to be kept. The robot offloads the calculation of the distances of every job from it and the distances of the spaces from the jobs and the sequence of the jobs and the spaces where it is to be kept. The cloud calculates the sequence of the jobs to be picked up by the robot and the location of the space where each job is to be kept. Cloud computing minimizes the memory of the robot as the database, which may be very voluminous, are kept in the Cloud. It also reduces the need of the robot for high processor as the complex calculation is to be done in the cloud.*

Keywords: *Automated Robots; Cloud environment; Database; Queries; Overhead.*

Date of Submission: 17-07-2017 Date of acceptance: 05-08-2017

I. Introduction

The launching of the Internet in the 1990s led to the limited sharing of resources. The Cloud Computing is an application of such a resource sharing mainly software rather than a hardware. The hardware cannot be shared over the internet but load on the hardware can be reduced by offloading the complex calculations to the cloud. It relieves remote devices from the burden of carrying out extensive computations [1]. J. M. Rabaey et al[5] spoke about energy optimal execution policy for a cloud-assisted mobile application platform. Cloud robotics uses the idea and includes the possibility of reducing the hardware requirement of a robot by storing the data in the cloud and getting them as and when required by querying them. It can also offload the complex calculations to the cloud and can query the result of the calculations as and when needed. This minimizes the hardware requirement [2].The robot simply receives help from the cloud [3] for doing its own work. R. Arumugam et al[4] spoke about the Cloud Computing framework for service robots. A.Kar et al[6] talked about the transportation of jobs by robots after the calculation of the optimized route by the cloud. A.Kar et al[7] talked about the transportation of jobs by robots according to priority after the calculation of the priority optimized route by the cloud.

II. The Problem Definition

The robot is assigned to pick up the jobs and keep them in the spaces provided. The robot receives the coordinate of the stick to be picked up. It reaches the coordinate and picks up the job. It then receives the coordinate of the shelf where this job is to be kept. It reaches the shelf and keeps the job. The robot then receives the coordinate of the next job to be picked up and it reaches the coordinate to pick up the next job and so on.
Let us suppose that the name of the robot is R.

ASSUMPTIONS

- There are four jobs denoted by Js to be kept in the shelf containing four spaces denoted by Ss.
- The jobs are scattered over the area.
- Any job may be picked up at any time.
- A particular job is to be kept at a particular space.
- R(0,0) is the initial coordinate of the robot.

As the jobs are picked up, the location of the robot is the location of the job that is just now picked up. As each job is kept in the space the location of the robot then becomes the location of the space, where the job is kept just now. This goes on till all the jobs are delivered.
Given below is the architecture of the floor where the robot is assigned to do the job.

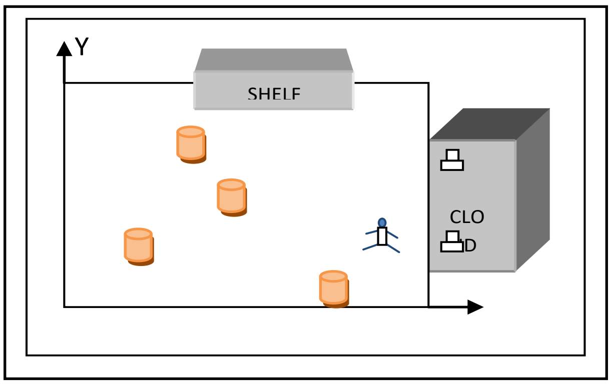

Fig – I: Floor where the Robot is operating

III. Dataset-I

a. ALGORITHM – I: PHASE BY PHASE SHORTEST PATH

The initial position of R is (0,0).
Let us suppose that the coordinates of the points where the jobs are scattered taken randomly are tabulated below:

TABLE – I: COORDINATES OF THE JOBS

Job_No.	Job_x	Job_y
J1	99	77
J2	9	6
J3	70	30
J4	88	62

Let us also take the positions of the spaces where the jobs are to be arranged, taken randomly are tabulated below:-

TABLE – II: COORDINATES OF THE SPACES

Space_No.	Space_x	Space_y
S1	20	100
S2	40	100
S3	60	100
S4	80	100

The position of the jobs and the spaces are represented in the figure below:-

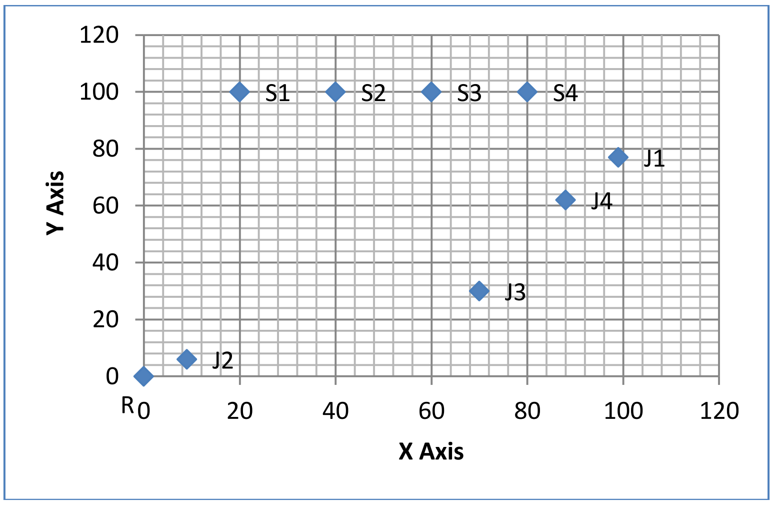

Fig – II: Position of the jobs and the spaces

The cloud calculates the distance of its initial position from each of the jobs and calculates the job that is nearest to the robot.
The distance calculated are given below:-
Let us suppose that the distance of the robot from each of the jobs are d1, d2, d3, d4 where
$d1 = \sqrt{((0-11)^2 + (0-47)^2)} = 48.27$
Similarly, d2, d3 and d4 are calculated and are given in the table below:
The robot, the jobs and the spaces are taken as the node of a graph and the distances from each other is given in the table below:-

TABLE – III: THE DISTANCES OF THE ROBOT FROM THE JOBS AND SPACES

Phase-I	Distance R-Ji-Si	Phase-II	Distance Si-Jj-Sj	Phase-III	Distance Si-Jj-Sj	Phase-IV	Distance Si-Jj-Sj	Total Distances
R-J1-S1	207.70	S2-J1-S1	145.60	**S4-J1-S1**	**112.11**			
R-J2-S2	**109.80**							
R-J3-S3	146.87	S2-J3-S3	146.87	S4-J3-S3	141.42	**S1-J3-S3**	**156.73**	
R-J4-S4	146.48	**S2-J4-S4**	**100.05**					
Phase Distances	**109.80**		**100.05**		**112.11**		**156.73**	**478.70**

where R denotes the robot, J1, J2, J3 and J4 are the jobs and S1, S2, S3 and S4 are the spaces.
Among the various paths available in the first Phase-I, the distance from R to J2-S2 is lowest, it is chosen. The job-space J2-S2 is exhausted, it will not be there in Phase-II.
In Phase-II, J4-S4 combination is lowest from S2 and so it is chosen. The job-space J4-S4 is exhausted and so it will not be there in Phase-III.
Phase-III has J1-S1 combination as the lowest from S4 and so it is chosen. The job-space J1-S1 is exhausted and so it will not be there in Phase-IV.
Phase-IV has only one alternative i.e. the combination J3-S3.
This completes the path and the total distance covered is 109.80+100.05+112.11+156.73 = **478.70**

We moved with the shortest path at every stage and added it to the total path. We started with the job that was at the minimum distance from the robot stationed at the origin. Thus we moved on to find all the spaces where the jobs are to be kept.
Although we wanted to find the route that covers minimum distance from the starting pointing at R(0,0) to the last S, but this algorithm cannot give any assurance that this path is the shortest. However this algorithm may not work well if the number of job and spaces are large.For finding out the first point, we have to scan n points. For the second, we have to scan (n-1) points and so on.
Total we have to scan $1+2+\ldots\ldots.+ (n-1) + n = n(n+1)/2 = ½.n^2 + ½.n \approx O(n^2)$ and hence the time complexity of this algorithm is $O(n^2)$.

b. ALGORITHM – II: OVERALL MINIMUM DISTANCE

From R, any job may be reached, but spaces cannot be reached. The minimum distance of R from all the jobs can be obtained from Table V, which is again given as under:-

TABLE – V: THE DISTANCES OF THE JOBS, SPACES AND THE ROBOT FROM EACH OTHER

	J1	J2	J3	J4
R	125.42	**10.82**	76.16	107.65
J1-S1		176.92	176.92	176.92
J2-S2	162.30		162.30	162.30
J3-S3	115.99	115.99		115.99
J4-S4	68.67	156.63	109.54	

We calculate the minimum distance in the matrix which is 10.82 i.e. R to J2. The row R and J2 column are exhausted.

TABLE – VI: THE DISTANCES OF THE JOBS, SPACES AND THE ROBOT FROM EACH OTHER

	J1	J2	J3	J4
R		**10.82**		
J1-S1			176.92	176.92
J2-S2	162.30		162.30	162.30
J3-S3	115.99			115.99
J4-S4	**68.67**		109.54	

The next minimum is 68.67 which is J4-S4 to J3. So J4-S4 row and J1 column are exhausted.

TABLE – VII: THE DISTANCES OF THE JOBS, SPACES AND THE ROBOT FROM EACH OTHER

	J1	J2	J3	J4
R		**10.82**		
J1-S1			176.92	176.92
J2-S2			162.30	162.30
J3-S3				**115.99**
J4-S4	**68.67**			

The next minimum is 115.99 which are J3- S3 to J4. So J3-S3 row and J4 column are exhausted.

TABLE – VII: THE DISTANCES OF THE JOBS, SPACES AND THE ROBOT FROM EACH OTHER

	J1	J2	J3	J4
R		**10.82**		
J1-S1			176.92	
J2-S2			**162.30**	
J3-S3				**115.99**
J4-S4	**68.67**			

The next minimum is 162.30 which is J2-S2 to J3. So J2-S2 row and J3 column are exhausted.

TABLE – IX: THE DISTANCES OF THE JOBS, SPACES AND THE ROBOT FROM EACH OTHER

	J1	J2	J3	J4
R		**10.82**		
J1-S1				
J2-S2			**162.30**	
J3-S3				**115.99**
J4-S4	**68.67**			

The path is R-J2-S2-J3-S3-J4-S4-J1-S1 and the total distance is 10.82 + 94.64 + 76.16 + 76.16 + 47.20 + 47.20 + 29.83 + 101.64 = 454.81. The result is significantly better than the Algorithm-I which was 478.70
If there are n jobs and n spaces, the time complexity of finding the minimum of the matrix takes $O(n^2)$ time.

c. COMPARISON OF THE TWO ALGORITHMS

Of the two algorithms discussed above, the complexity of calculation is same. The distance to be covered by the robot for doing the whole job in the 1st algorithm is 478.70 whereas in case of the 2nd algorithm it is 454.81. This is for the above taken datasets.

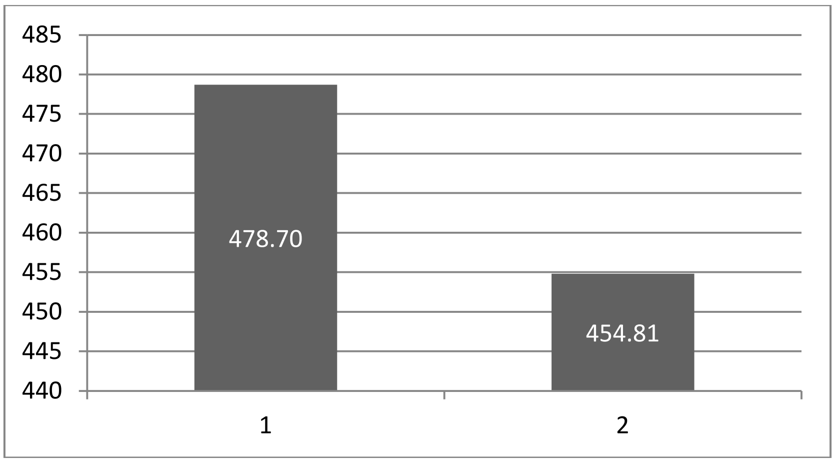

Fig – III: Comparison of the two Algorithms

The problem can be taken as Traveling Salesman Problem (TSP) with an additional constraint that one S is to be followed by one J and vice versa.

IV. Dataset – II

d. ALGORITHM – I: PHASE BY PHASE SHORTEST PATH

The initial position of R is (0,0).
Let us suppose that the coordinates of the points where the jobs are scattered taken randomly are tabulated below:

TABLE – I: COORDINATES OF THE JOBS

Job_No.	Job_x	Job_y
J1	56	88
J2	76	68
J3	50	58
J4	40	19

Let us also take the positions of the spaces where the jobs are to be arranged, taken randomly are tabulated below:-

TABLE – II: COORDINATES OF THE SPACES

Space_No.	**Space_x**	**Space_y**
S1	**20**	**100**
S2	**40**	**100**
S3	**60**	**100**
S4	**80**	**100**

The position of the jobs and the spaces are represented in the figure below:-

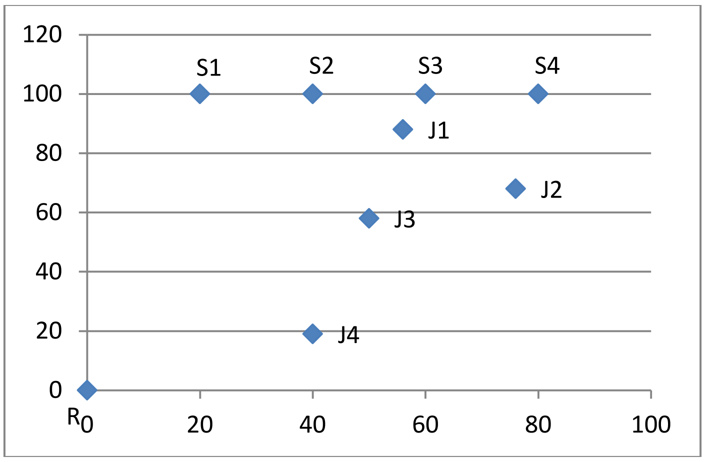

Fig – III: Position of the jobs and the spaces

The cloud calculates the distance of its initial position from each of the jobs and calculates the job that is nearest to the robot.
The distance calculated are given below:-

TABLE – III: THE DISTANCES OF THE ROBOT FROM THE JOBS AND SPACES

Phase-I	Distance R-Ji-Si	Phase-II	Distance Si-Jj-Sj	Phase-III	Distance Si-Jj-Sj	Phase-IV	Distance Si-Jj-Sj	Total Distances
R-J1-S1	142.25	**S3-J1-S1**	**50.60**					
R-J2-S2	150.15	S3-J2-S2	83.94	**S1-J2-S2**	**112.66**			
R-J3-S3	**119.75**							
R-J4-S4	134.62	S3-J4-S4	173.77	S1-J4-S4	173.77	**S2-J4-S4**	**171.34**	
Phase Distances	**119.75**		**50.60**		**112.66**		**171.34**	**454.35**

where R denotes the robot, J1, J2, J3 and J4 are the jobs and S1, S2, S3 and S4 are the spaces.
Among the various paths available in the first Phase-I, the distance from R to J3-S3 is lowest, it is chosen. The job-space J3-S3 is exhausted, it will not be there in Phase-II.
In Phase-II, J1-S1 combination is lowest from S3 and so it is chosen. The job-space J1-S1 is exhausted and so it will not be there in Phase-III.

Phase-III has J2-S2 combination as the lowest from S4 and so it is chosen. The job-space J2-S2 is exhausted and so it will not be there in Phase-IV.
Phase-IV has only one alternative i.e. the combination J4-S4.
This completes the path and the total distance covered is 119.75+50.60+112.66+171.34 = **454.35**
We moved with the shortest path at every stage and added it to the total path. We started with the job that was at the minimum distance from the robot stationed at the origin. Thus we moved on to find all the spaces where the jobs are to be kept.
Although we wanted to find the route that covers minimum distance from the starting pointing at R(0,0) to the last S, but this algorithm cannot give any assurance that this path is the shortest. However this algorithm may not work well if the number of job and spaces are large. For finding out the first point, we have to scan n points. For the second, we have to scan (n-1) points and so on.
Total we have to scan 1+2+……..+ (n-1) + n = n(n+1)/2 = ½.n^2 + ½.n ≈ O(n^2) and hence the time complexity of this algorithm is O(n^2).

e. ALGORITHM – II: OVERALL MINIMUM DISTANCE

From R, any job may be reached, but spaces cannot be reached. The minimum distance of R from all the jobs can be obtained from Table V, which is again given as under:-

TABLE – V: THE DISTANCES OF THE JOBS, SPACES AND THE ROBOT FROM EACH OTHER

	J1	J2	J3	J4
R	104.31	101.98	76.58	**44.28**
J1-S1		102.45	89.56	121.38
J2-S2	68.17		91.34	129.17
J3-S3	55.82	78.95		126.61
J4-S4	117.17	122.59	141.95	

We calculate the minimum distance in the matrix which is 44.28 i.e. R to J4. The row R and J4 column are exhausted.

TABLE – VI: THE DISTANCES OF THE JOBS, SPACES AND THE ROBOT FROM EACH OTHER

	J1	J2	J3	J4
R				**44.28**
J1-S1		102.45	89.56	
J2-S2	68.17		91.34	
J3-S3	**55.82**	78.95		
J4-S4	117.17	122.59	141.95	

The next minimum is 55.82 which is J3-S3 to J1. So J3-S3 row and J1 column are exhausted.

TABLE – VII: THE DISTANCES OF THE JOBS, SPACES AND THE ROBOT FROM EACH OTHER

	J1	J2	J3	J4
R				**44.28**
J1-S1		102.45	**89.56**	
J2-S2			91.34	
J3-S3	**55.82**			
J4-S4		122.59	141.95	

The next minimum is 89.56 which are J1- S1 to J3. So J1-S1 row and J3 column are exhausted.

TABLE – VII: THE DISTANCES OF THE JOBS, SPACES AND THE ROBOT FROM EACH OTHER

	J1	J2	J3	J4
R				**44.28**
J1-S1			**89.56**	
J2-S2				
J3-S3	**55.82**			
J4-S4		**122.59**		

The next minimum is 122.59 which is J4-S4 to J2. So J4-S4 row and J2 column are exhausted.

The path is R-J4-S4-J2-S2. Then the path breaks and there is a loop of J3-S3-J1-S1 or J1-S1-J3-S3.
The total distance, if we take the path R-J4-S4-J2-S2=J1-S1-J3-S3 is 367.77.
The total distance, if we take the path R-J4-S4-J2-S2=J3-S3-J1-S1 is 351.98.
Both the results are significantly better than the Algorithm-I which was 454.35. But the time complexity of the calculation is more than $O(n^2)$, as we did not get a definite path. We can say the lower bound is n^2 i.e. $\Omega(n^2)$.

f. COMPARISON OF THE TWO ALGORITHMS

Of the two algorithms discussed above, the complexity of the 1st algorithm is $O(n^2)$ and the 2nd algorithm is $\Omega(n^2)$. In terms of complexity of calculation the 1st algorithm is better. But as far as the distance covered, the 2nd algorithm provides better result. The distance to be covered by the robot for doing the whole job in the 1st algorithm is 454.35 whereas in case of the 2nd algorithm it is either 367.77 or 351.98. This is for the above taken datasets.

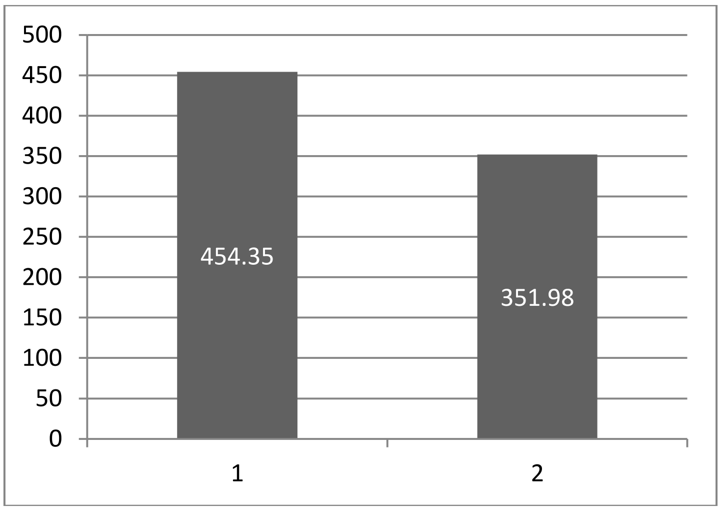

Fig – III: Comparison of the two Algorithms

The problem can be taken as Traveling Salesman Problem (TSP) with an additional constraint that one S is to be followed by one J and vice versa.

V. Cost Benefit Analysis

The idea of the research was to make the work done by a robot taking help from the cloud in case of storage and complex calculations. This reduces the need for high memory capacity of the robot as the robot only need to store the coordinate that it is to go and after reaching the coordinate it may overwrite the memory with the next coordinate that it is to go. There is no need to store the coordinate where it has already visited as all the coordinates are there in the cloud. The whole calculations are done in the cloud. So the need for high processor for calculation purpose is also minimized as it does not need to do any calculations. It can get the result of any calculations from the cloud. The reduction for the need of high memory capacity and high processor for calculation may lead to reduction of the complexity of the processes running inside the robot. It may reduce the situations of process starvation and process deadlock and may reduce the price of the robot.

But the operations to be done in the cloud have a cost. If this cost is sufficiently less than that of the cost of the processor and memory that the robot has to utilize had the calculations been done by the robot itself, then it is feasible to use the memory and processor of the Cloud, otherwise it is infeasible.
Let the cost of per unit calculations done by the Cloud be C.
If there are p jobs to be kept at p spaces then total cost of calculations of the Cloud = $p^2 * C$
Let the cost of the present processor and memory of the robot be x.
Let the cost of the processor and memory required for the robot had the calculations been done by the robot be X.
The calculations to be done in the Cloud is feasible if

$$\boxed{P^2 * C + x < X}$$

VI. Conclusion

Whatever may be the algorithm that the cloud uses, it sends the end results to the robot. When the robot is in the point R(0,0), the robot queries the cloud about the next location to go. The cloud replies the robot to go to J2 giving the coordinate of J2. As soon as the robot moves to J2 and picks up the job, it keeps the job at S2. Then the robot queries the cloud about the next job. The cloud replies about the job along with its location that is next in the path already calculated by the cloud and so on. This goes on until all the jobs are kept in the spaces i.e. all the jobs are over.

The idea of the research was to make the work done by a robot taking help from the cloud in case of storage and complex calculations. This reduces the need for high memory capacity of the robot as the robot only need to store the coordinate that it is to go and after reaching the coordinate it may overwrite the memory with the next coordinate that it is to go. There is no need to store the coordinate where it has already visited as all the coordinates are there in the cloud. The whole calculations are done in the cloud. So the need for high processor of the robot for calculation purpose is also minimized as it does not need to do any calculations. It can get the result of any calculations from the cloud.

The reduction for the need of high memory capacity and high processor for calculation may lead to reduction of the complexity of the processes running inside the robot. It may reduce the situations of process starvation and process deadlock and may reduce the price of the robot.

References

[1]. P. Mell, T. Grance, The NIST Definition of Cloud Computing, National Institute of Standards and Technology Special Publication 800-145, 2011.

[2]. E. Guizzo, Robots with Their Heads in the Clouds, IEEE Spectrum, March 2011.

[3]. E. Guizzo, Cloud Robotics: Connected to the Cloud Robots Get Smarter, IEEE Spectrum, January 2011.2012 Florida Conference on Recent Advances in Robotics 6 Boca Raton, Florida, May 10-11, 2012

[4]. R. Arumugam, V. R. Enti, L. Bingbing, W. Xiaojun, K. Baskaran, F. F. Kong, A. S. Kumar, K. D. Meng, G. W. Kit, M. Rakotondrabe, and I. Ivan, "Davinci: A cloud computing framework for service robots."International Conference on Robotics and Automation, 2010, pp. 3084– 3089.

[5]. J. M. Rabaey, Ed., Digital Integrated Circuits. Prentice Hall, 1996. Y. Wen, W. Zhang, K. Guan, D. Kilper, and H. Luo, "Energy-optimal execution policy for a cloud-assisted mobile application platform," NTU Technical Report, 2011.

[6]. A.Kar, A. K. Dutta, and S. K. Debnath (2017), Task management of Robot using Priority Job Scheduling, International Journal of Computer Science and Information Technology vol 8(2) pg 260-265.

[7]. A.Kar, A. K. Dutta, and S. K. Debnath (2016b), Task management of robot using Cloud Computing. IEEE International Conference on Computer, Electrical and Communication Engineering, 2016.