Actor-Actor Coordination in Wireless Sensor Actor Network Using Row Index Data Access Matrix Technique

Rida Hudaya\textsuperscript{1,2,*} and Cucun Wida Nurhaeti \textsuperscript{1,3}

\textsuperscript{1}Departement of Electrical Engineering Politeknik Negeri Bandung, Bandung 40012, Indonesia
\textsuperscript{2}Head of Instrumentation & Control Systems Expertise Group, Bandung State Polytechnic, Bandung 40012, Indonesia
\textsuperscript{3}Head of Manufacture & Industrial Electronic Expertise Group, Bandung State Polytechnic, Bandung 40012, Indonesia

Abstract. The use of Row Indexed Data Access Matrix technique in actor-actor coordination for Wireless Sensor Actor Networks provides an easy technique to manage the large amounts of data and signals electric. It also provides convenience in adding or reducing the number of sensors, actuator, actors and sink node. This product research has been tested on the real plant. The plant used is a digital precision farming system that is conditioned in a greenhouse. Actor-actor coordination works using rida-M technique at actor node and sink node. Interaction between actor nodes is coordinated by sink node. The system testing has been performed on the real system and the actor-actor coordination configuration has Life time of power supply, 12 hours for sink node and 60 hours for sensor node.

1. Introduction
The experiment of the Wireless Sensor Actor Networks (WSAN) configuration used the structure as shown in Figure 1. A group of sensors is coordinated by an actor. All of the actors are connected to the Sink Node. The sensor-actor and actor-sink communication uses 2.4 GHz radio transceiver. This configuration has been tested and applied to support the digital precision farming in agricultural technology. The sensors are realized in several forms such as humidity sensors, temperature sensors, total dissolved solid sensors, light intensity sensors, moisture sensors, wind speed sensors and sun position sensors.

WSAN is a group of sensors and actors linked by wireless medium to perform distributed sensing and acting tasks. Sensor–actor and actor–actor coordination problems are a research challenges\textsuperscript{1}.

In WSAN, to provide effective sensing and acting, a distributed local coordination mechanism is necessary among sensors and actors. The actor node must have short delays, low energy consumption and proper task assignment\textsuperscript{2}.

2. Previous Researchs
The application of WSAN in a networked control system creates an unavoidable delay which can lead to system instability. Through reference-model technique, stability can be maintained\textsuperscript{3,4,5,6}.
The monitoring technology in WSAN using a big number of signal in the real time systems can be solved by using Row Index Data Access Matrix (rida-M) technique\(^7\). Implementation of rida-M technique is not limited to monitoring systems, but also at the level of homogeneous/heterogeneous analog/digital data interface software\(^8,9\).

The Communication between node in WSAN using the command/response protocols is best suited for the sensor-actor level protocol\(^10,11\). A multi-hop routing mobile data collection algorithm in actor and sink node is proposed based on dynamic polling point selection with delay constraints\(^12\).

3. Method Proposed

3.1. Data in the Actor Node dan Sink Node.

In Figure 1, each Actor connected with a number of Sensors. The sensor \(S_{11}\) produces the data \(d_{sensor11}\), the first sensor connected to \(Actor_1\), and the data store on the \(Actor_1\), and so on, up to the sensor \(S_{nn}\) which generates the data \(d_{sensornn}\), the data from the \(n\)th sensor connected to the \(Actor_n\) and the data is stored in the \(Actor_n\). If \(D_{sensor\_actor}\) is the sensor data matrix in the Actor Node, then for each Actor Node can be expressed as follows:

\[
D_{sensor\_actor1} = \begin{bmatrix}
    d_{sensor11} \\
    d_{sensor12} \\
    d_{sensor13} \\
    \vdots \\
    d_{sensor1n}
\end{bmatrix}
\]

\[
D_{sensor\_actorn} = \begin{bmatrix}
    d_{sensor1n} \\
    d_{sensor2n} \\
    d_{sensor3n} \\
    \vdots \\
    d_{sensornn}
\end{bmatrix}
\]

If \(D_{sensor\_actor\_sink}\) is a sensor-actor-data-matrix in Sink Node and \(D_{sensor\_sink}\) is the sensor-sink-data-matrix in Sink Node, then the sensor-data-matrix of Eq. (1) can be expressed as follows:
\[ D_{\text{sensor\_actor\_sink}} = \begin{bmatrix} D_{\text{sensor\_actor1}} \\ D_{\text{sensor\_actor2}} \\ D_{\text{sensor\_actor3}} \\ \vdots \\ D_{\text{sensor\_actorn}} \end{bmatrix} \] (2)

or we can express also as follows:

\[ D_{\text{sensor\_sink}} = \begin{bmatrix} d_{\text{sensor11}} \\ \vdots \\ d_{\text{sensor1n}} \\ d_{\text{sensor21}} \\ \vdots \\ d_{\text{sensor2n}} \\ d_{\text{sensor31}} \\ \vdots \\ d_{\text{sensor3n}} \\ \vdots \\ d_{\text{sensor1}} \\ \vdots \\ d_{\text{sensor\_sink}} \end{bmatrix} \] (3)

In Figure 1, each Actor is connected to a number of Actuators. Data \( d_{\text{actuator11}} \) comes from \( \text{Actor}_1 \) for actuator \( A_{11} \) which is the first actuator connected with \( \text{Actor}_1 \) and the data is stored in the \( \text{Actor}_1 \), and so on, up to data \( d_{\text{actuatornn}} \), the data data coming from the \( n \)th actor for the actor connected to the \( \text{Actor}_n \) and the data is stored in \( \text{Actor}_n \). If \( D_{\text{actuator\_actor\_sink}} \) is a actuator-actor-data-matrix in Sink Node and \( D_{\text{actuator\_sink}} \) is the actuator-sink-data-matrix in Sink Node, then the actuator-data-matrix of Eq. (4) can be expressed as follows:

\[ D_{\text{actuator\_actor1}} = \begin{bmatrix} a_{d11} \\ a_{d12} \\ a_{d13} \\ \vdots \\ a_{d1n} \end{bmatrix} \]...

\[ D_{\text{actuator\_actorn}} = \begin{bmatrix} a_{dn1} \\ a_{dn2} \\ a_{dn3} \\ \vdots \\ a_{dnn} \end{bmatrix} \] (4)

If \( D_{\text{actuator\_actor\_sink}} \) is a actuator-actor-data-matrix in Sink Node and \( D_{\text{actuator\_sink}} \) is the actuator-sink-data-matrix in Sink Node, then the actuator-data-matrix of Eq. (4) can be expressed as follows:

\[ D_{\text{actuator\_actor1}} = \begin{bmatrix} d_{\text{actuator11}} \\ d_{\text{actuator12}} \\ d_{\text{actuator13}} \\ \vdots \\ d_{\text{actuator1n}} \end{bmatrix} \]
\[
D_{\text{actuator}_{\text{actorn}}} = \begin{bmatrix}
D_{\text{actuator}11} \\
D_{\text{actuator}12} \\
\vdots \\
D_{\text{actuator}1n} \\
D_{\text{actuator}21} \\
\vdots \\
D_{\text{actuator}2n} \\
\vdots \\
D_{\text{actuator}n1} \\
\vdots \\
D_{\text{actuator}nn}
\end{bmatrix}
\] (5)

or we can express also as follows:

\[
D_{\text{actuator}_{\text{sink}}} = \begin{bmatrix}
D_{\text{actuator}11} \\
\vdots \\
D_{\text{actuator}1n} \\
D_{\text{actuator}21} \\
\vdots \\
D_{\text{actuator}2n} \\
\vdots \\
D_{\text{actuator}31} \\
\vdots \\
D_{\text{actuator}3n} \\
\vdots \\
D_{\text{actuator}n1} \\
\vdots \\
D_{\text{actuator}nn}
\end{bmatrix}
\] (6)

Based on Equation (3) and Equation (6), we can say that \(D_{\text{sensor}_{\text{sink}}}\) and \(D_{\text{actuator}_{\text{sink}}}\) are Actor-Actor coordination matrices.

From Equation (1), Equations (2), Equations (4) and Equation (5) then the matrices in each Actor Node and Sink Node can each be expressed using the rida-M technique as follows:

\[
D_{\text{ridaM}_{\text{actor}}} = \begin{bmatrix}
D_{\text{sensor}_{\text{actor}}} \\
D_{\text{actuator}_{\text{actor}}}
\end{bmatrix}
\] (7)

\[
D_{\text{ridaM}_{\text{sink}}} = \begin{bmatrix}
D_{\text{sensor}_{\text{sink}}} \\
D_{\text{actuator}_{\text{sink}}}
\end{bmatrix}
\] (8)

Equation 7 and Equation 8 show the characteristics of the rida-M technique. The placement of data at each node is arranged downward linearly. This shows the placement of data in the memory arrangement in such sequence and executed one by one sequentially as well.

3.2. Row Indexed Data Access Matrix Algorithm.

The application of rida-M algorithm can be applied by using Equation 7 and Equation 8. Equation 7 is algorithm for data process in Actor Node and Equation 8 is algorithm for data process in Sink Node.

The line position in the matrix of Equation 7 and Equation 8 are a sequential index from top to bottom. The index is used to designate the sensor and actuator address. This is because the sensor and actuator address line has the same index as the variable name of a sensors. The following sub-section will explain more technically to use the index of the line.

Sensor-Actor Node Data Transfer.

The process of Sensor-Actor Node Data Transfer between the Sensor Node \(S_{11}\) and the Actor Node \(Actor_{1}\) can be done with the steps as shown in the following algorithm:

1. Declare variable \(d_{\text{sensor}11}\) until \(d_{\text{sensor}1n}\) and variable \(d_{\text{actuator}11}\) until \(d_{\text{actuator}1n}\) in sequence.
2. Read the data from $S_{11}$ and save it in variable $d_{sensor11}$.
3. Read data from variable $d_{actuator11}$ and send to actuator $A_{11}$.
4. The same way with the step 2 then step 3, do for the next sensor until $S_{1n}$ and also for actuator until $d_{actuator1n}$.
5. Repeat steps 2,3,4 for the next Actor Node until $Actor_n$.

**Actor-Sink Node Data Transfer.**

The process of Actor-Sink Node Data Transfer between the Actor Node $A_i$ and Sink Node can be done with the steps as shown in the following algorithm:
1. Declare variable $d_{sensor11}$ until $d_{sensor nn}$ and variable $d_{actuator11}$ until $d_{actuator nn}$ in sequence.
2. Read the data from $Actor_i$ and save it in variables $d_{sensor11}$ until $d_{sensor1n}$.
3. Read the data from $d_{actuator11}$ until $d_{actuator1n}$ and send it to $Actuator_i$.
4. The same way with the step 2 then step 3, do for the next actor until $Actor_n$.

Actor-actor communication is done through Sink Node and the data Sink Node then passed to database server.

**4. Experimental Result**

The following tests were performed under arduino uno microcontroller module environment, using two different types of baud rate, 9600 and 115200 bps for 1,10,20,30,40,50,60,80,100,200,400 and 600 sensors.

The example of code is shown in Figure 2. This program code processes 10 sensors, with the baud rate 115200 bps and the test result for all tests is shown in Figure 3. The ratio of the execution time of each number of sensors between two different bit rates is proportional to its baud rate. The execution time exponentially changes with the number of sensors.

The variables in line 11 until 20 in Figure 2 used to save the data sensors and in this case, represent application of rida-M technique or Equation 1. The time of execution is performed by code in line 31,53,54. The data transfer process, from sensor node to actor node is performed in line 32 until 51.

In the same way we can transfer data from actor node to sink node (Equation 3) with code and execution results similar to Figures 2 and 3. In this case, the sink node has a function as actor-actor coordination. Using protocol slave-master communication, the sink node coordinates actor nodes. So actor-actor coordination, exactly use the same technique as use by sensor-actor node which performed by sink node as master coordinator for actor nodes.

```
01 //This program is used for 115200 bps
02 //and 10 sensors.
03
04 #include <Time.h>
05
06 unsigned long prevTime;
07 unsigned long nowTime;
08 unsigned long Time;
09
10 //Variables to save data sensors
11 int dsensor01;
12 int dsensor02;
13 int dsensor03;
14 int dsensor04;
15 int dsensor05;
16 int dsensor06;
17 int dsensor07;
18 int dsensor08;
```
19 int dsensor09;
20 int dsensor10;
21
22 void setup() {
23 // Set serial communication to 115200 bps:
24 Serial.begin(115200);
25 }
26
27 void loop() {
28   // put your main code here, to run repeatedly:
29   Serial.println();
30   Serial.println("=".repeat(25));
31   prevTime = micros();
32   Serial.print(1); // Data sensor received via serial comm.
33   dsensor01=1; // Data sensor received is saved in a variable
34   Serial.print(1);
35   dsensor02=1;
36   Serial.print(1);
37   dsensor03=1;
38   Serial.print(1);
39   dsensor04=1;
40   Serial.print(1);
41   dsensor05=1;
42   Serial.print(1);
43   dsensor06=1;
44   Serial.print(1);
45   dsensor07=1;
46   Serial.print(1);
47   dsensor08=1;
48   Serial.print(1);
49   dsensor09=1;
50   Serial.print(1);
51   dsensor10=1; // Data sensor for sensor 10th
52   "=".repeat(25);
53   nowTime = micros();
54   Time=nowTime-prevTime;
55   Serial.println();
56   Serial.println("Execution time = ");
57   Serial.println(Time);
58 }

Figure 2. List of Code under Arduino Uno microcontroller module environment.

The following test is the result of data acquisition of the sensor on each sensor node. The test is done in the field in a greenhouse as shown in Figure 4 and Figure 5 to support IoT-based digital precision farming application.

The experiment was carried out using a powerbank source to study the effect of rida-M applications on the power used. The current consumption used by a sink node when sending and receiving data continuously for 11.8 hours is 434 mA. This shows that the powerbank capacity is 5200 mAh, as shown in Equation (8) and Equation (9)

...
From the measurement result on the sink node:

\[ I_{\text{out, sink node}} = 434 \text{ mA} \]
\[ t_{\text{sink node}} = 11.8 \text{ jam} \]
\[ \text{Capacity power bank} = I_{\text{out, sink node}} \times t_{\text{sink node}} \]
\[ = 5121.2 \text{ mAh} \quad (9) \]

From the measurement result on the sensor node:

\[ I_{\text{out, sensor node}} = 88 \text{ mA} \]
\[ t_{\text{sensor node}} = 58.5 \text{ jam} \]
\[ \text{Capacity power bank} = I_{\text{out, sensor node}} \times t_{\text{sensor node}} \]
\[ = 5148 \text{ mAh} \quad (10) \]

So life time of power supply, 12 hours for sink node and 60 hours for sensor node.

5. Conclusion

Based on the activities that have been carried out in this research, several conclusions can be taken as follows:

- Actor-actor coordination works using rida-M technique at actor node and sink node. Interaction between actor nodes is coordinated by sink node.
The system testing has been performed on the real system and the actor-actor coordination configuration has Life time of power supply, 12 hours for sink node and 60 hours for sensor node.

Acknowledgments
This work was supported by Politeknik Negeri Bandung under research grant.

References
[1] Ian F. Akyildiz, Ismail H. Kasimoglu. Wireless Sensor and Actor Networks: Research Challenges, Ad Hoc Networks, vol 2, no 4, pp351-367, Oct. 2004.
[2] Elmazi Donald, Kulla Elis, Matsuo Keita, Oda Tetsuya, Spaho Evjola, Barolli Leonard. A mobility-aware fuzzy-based system for actor selection in wireless sensor–actor networks, Journal of High Speed Networks, vol. 21, no. 1, pp. 15-25, 2015.
[3] Nathan B. Loden, J.Y. Hung, 2005, “An adaptive PID controller for network based control systems”, Industrial Electronics Society, 31st Annual Conference of IEEE, 2005.
[4] Rida Hudaya. A Reference-Model Controller to Mitigate The Effect of Time Delay Change in A Networked Control Systems, Jurnal Teknologi, E-ISSN 2180-3722, vol 77 No.22, 2016.
[5] Rida Hudaya. A Reference-Model Controller to Mitigate The Effect of Time Delay Change in A Network Control Systems”, in: Proceedings of the First International Conference on Engineering, Technology, and Applied Business 2014 (ICETAB 2014), Book of Proceedings Abstract pp 11, 4th December 2014.
[6] Rida Hudaya, Feriyonika, Cucun Wida Nurhaeti. Adaptive Retuning PID to overcome effect of delay change in networked control systems, Proceedings of the Fourth Industrial Research Workshop and National Seminar 2013 (IRWINS 2013), Book of Proceedings pp 145-150, 20th November 2013.
[7] Rida Hudaya, Cucun Wida Nurhaeti. Application of Row Index Data Access Matrix Algorithm in HMI/Scada Systems, Proceedings of the First PIKSI International Conference on Knowledge and Sciences 2014 (PIKSI 2014), Book of Proceedings Abstract pp 58, 18-19th November 2014.
[8] Rida Hudaya, 2014, Application of row index data access matrix algorithm on heterogeneous command/status for digital data interface software systems, in: Proceedings of the Fifth Industrial Research Workshop and National Seminar 2014 (IRWINS 2014), Book of Proceedings Abstract pp 34, 12-13th November 2014.
[9] Rida Hudaya, 2013, Application of row index data access matrix algorithm on homogeneous command/status for digital data interface software systems, in: Proceedings of the Fourth Industrial Research Workshop and National Seminar 2013 (IRWINS 2013), Book of Proceedings pp 158-161, 20th November 2013.
[10] Rida Hudaya, 2012, The study of the syntax constructs of command / respond protocols with ASCII characters on an analog input multidrop module network, Elektran Jurnal Terapan Teknik Elektro Vol 2 No 2, pp 34-39, ISSN 2089-3116, December 2012.
[11] Rida Hudaya, 2012, Review the pattern of the command / respond protocols in the remote I / O digital module, TEDC Jurnal Ilmiah Berkala Vol 2 No 2, pp 81-85, ISSN 1978-0060, Mei 2012.
[12] Shuo Li, Jun Peng, Weirong Liu, Zhengfa Zhu, Kuo-Chi Lin. Sensor and Actuator Networks Based on Dynamic Polling Point Selection, Sensors 2014, 14, 95-116; doi:10.3390/s140100095, Open Access Sensors ISSN 1424-8220