Journal of Multidisciplinary Engineering Science and Technology (JMEST)
ISSN: 2458-9403
Vol. 7 Issue 1, January - 2020

Markov-based Model for Packet Delivery in Multi-Hop Wireless Sensor Networks Using R

Edward Kennedy
Dept. of Industrial Engineering,
New Mexico State University
Las Cruces, NM 88003, USA
ekennedy@nmsu.edu

Han-suk Sohn
Dept. of Industrial Engineering,
New Mexico State University
Las Cruces, NM 88003, USA
hsohn@nmsu.edu

***Abstract*— This paper presents an R-based Markov chain model for packet delivery in a multi-hop wireless sensor network and it was tested using the "markovchain" library. The model was a 5-node mesh network with a source node, destination node, and three intermediate sensor nodes. The model can be generalized to more complex mesh networks that use multi-hop routing. A simplified Markov decision process model was also presented, which has the form of a minimum total weighted cost node deployment policy dependent only upon the number of sensor nodes and their associated transmission range.**

Keywords—Markov model; multi-hop wireless sensor networks; R libraries; packet delivery

I. INTRODUCTION

A wireless sensor network (WSN) is a collection of widely-dispersed autonomous sensor nodes deployed collaboratively enabling physical measurements made by the sensor nodes to be collected, monitored, and processed for further use. The sensor nodes communicate wirelessly and are often equipped with computation capabilities. WSNs are a key feature in contemporary industrial and utility automation, supervisory control and data acquisition (SCADA) systems, smart grids, the Internet of Things (IoT) and a wide variety of other applications [1]. Fig. 1 depicts a nominal wireless sensor network. Many other network architectures are possible.

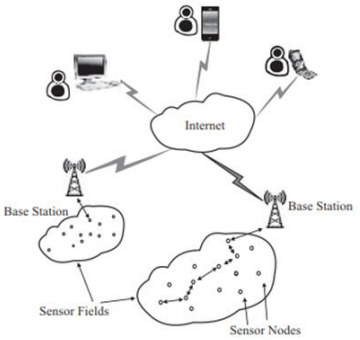

Fig. 1. An example of a wireless sensor network [1]

WSNs operate as stochastic systems because of randomness in the monitored environments [2]. Since the future state of a sensor node is generally dependent upon the current state and not past states, Markov decision processes can be used to model WSNs to help optimize the WSN's operations [2]. Several recently-developed variants of Markov decision processes were examined for applicability towards WSN packet delivery models. These included Markov blankets [3], imprecise Markov models [4], and updated hidden Markov models [5], [6]. All of these, especially hidden Markov models, were applicable towards WSN packet modeling but either added unneeded complexity or were challenging to implement in R.[1] Also considered, albeit briefly, were various decision-making algorithms used in multi-agent systems; these included Bayesian game theory, graph theory, and swarm intelligence [7].

The research presented here is intended to find an accurate Markov-based representation of data packet delivery for a generic WSM that was implementable in R and would inform a Markov decision model seeking an optimum policy for data packet delivery. A further research led to the narrowing of the problem by confining the WSN to multi-hop implementations and the use of a discrete time Markov chain to represent the network.

II. PROBLEM STATEMENT

Consider two widely used WSN topologies illustrated in Figure 2. The arrangement on the left is known as a star topology and uses single-hop communications. In a single-hop WSN, the sensor nodes transmit their measurements directly to the base station (i.e. in a single hop). This configuration requires power at the sensor node that is sufficient to enable the transmission. Since the sensor nodes are self-powered, often by a battery, care must be taken in selecting a transmission protocol, and associated duty cycle, that minimizes power consumption, especially where the transmission is over large geographic distances.

An alternative arrangement is the mesh topology using multi-hop communications as shown on the right side of Fig. 2. Here sensor nodes capture and disseminate their own data as well as serve as relays for other sensor nodes. In this arrangement, the data

[1] Hidden Markov models are an exception. There are several well-developed packages in R for hidden Markov models such as HMM (Hidden Markov Model) and depmixS4 (Dependent Mixture Models S4 Class).

packet is issued from the sensor node to the base station through a series on multiple intermediate hops. The power consumption is considerably less than in the single-hop case since the transmission distances for the individual hops are much smaller than the distances between the sensor nodes and the base station.

While less electrical power may be required by the sensor nodes in a multi-hop configuration, several new complications are introduced. Among these is the requirement that the nodes must now collaborate to enable their aggregate data to be organized and received by the base station [8]. The task of finding a multi-hop path from a sensor node to the base station is a complex routing problem; when a node serves as a relay for multiple routes it may analyze and pre-process sensor data in the network leading to the elimination of redundant information or aggregate the data such that it may be smaller than the original packet and as a consequence adjust the routing path for a more efficient packet delivery [1], [8].[2] Finding a solution to such problems is becoming increasingly important as the presence of WSNs becomes more ubiquitous in industry and everyday life.

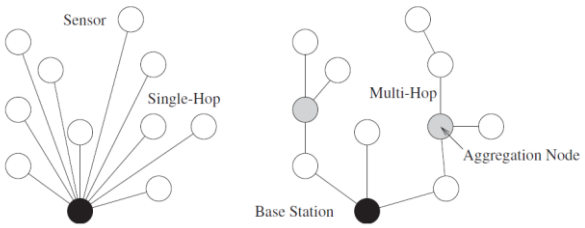

Fig. 2. Single-hop and multi-hop WSN communications [8]

Multi-hop WSNs have previously been modeled as discrete time Markov chains using both cooperative and non-cooperative automatic repeat request protocols [9].[3] The approach to be taken here is to model WSN packet delivery as a discrete time Markov chain and attempt to implement the model using existing libraries in R. Several libraries were evaluated with the search being confined to the CRAN repository.[4] A summary of their principal capabilities is shown in Table 1. This list is not exhaustive and Markov-related libraries related to specific applications, such as genomics, were not considered.

The R package "markovchain" was selected because it was built specifically to accommodate discrete time Markov chains and appeared more capable than DTMCPack; it is also capable of modeling continuous time Markov chains, as well as homogeneous and some forms of inhomogeneous Markov chains [10].

Table 1. Summary of principal Markov-related capabilities of the R libraries evaluated.

R Package	Discrete Time Markov Chain	Continuous Time Markov Chain	Hidden Markov Model	Markov Chain Monte Carlo
Markov chain	X	X		
HMM			X	
depmixS4			X	
mcmc				X
msm		X	X	
DTMCPack	X			

We may now summarize the problem statement as follows: use the "markovchain" library to implement a discrete time Markov chain model of a multi-hop WSN as a tool for aiding in the development of a Markov decision process to discover an optimum packet delivery policy for the multi-hop network.

III. Markov Decision Model Formulation

Markov decision models require that the future system state be dependent upon only the current state and not on any of the past states. The solution of the model, referred to as a policy, may be implemented in a WSN as a look-up table stored in the sensor node's memory [2]. Multi-hop WSNs have been modeled as a discrete time Markov chain with an absorbing state [9]. Following the techniques developed in [9] a multi-hop WSN, with cooperative automatic repeat request protocols, is represented as a Markov chain in figure 3. In our initial case, the WSN has five equidistant sensors with four links and a transmission range of R_t=2 units from each node. For the general case, we would have a source node (n=1) followed by any number of intermediate nodes (nodes 2, 3, and 4 in figure 3), and concluding with a destination (n=5 in Fig. 3 but generalizable to n = N). The final destination node, nominally the base station, does not retransmit the data packet and is represented as an absorbing state.

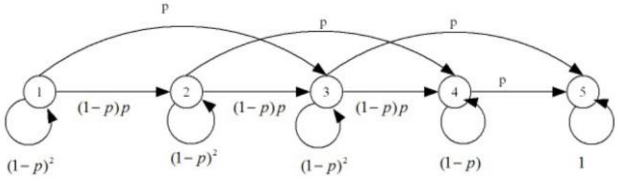

Fig. 3. Package transmission as a discrete time Markov chain model [9]

Let p represent the probability a packet is delivered correctly from node n to a node R_t links distant. The probability that the packet is not received by this node is therefore (1–p); if not received then, under the

[2] There is also an NP-complexity issue for large mesh networks because as more nodes are added the number of links increases exponentially.

[3] In cooperative automatic repeat request protocols the data packet is retransmitted by a neighboring node if the original data packet was not received. This retransmission does not occur for non-cooperative protocols. For this paper, cooperative automatic repeat protocols will be assumed with the packet being retransmitted by the adjacent (prior) node if the transmission from the original sending node was not received.

[4] The "Comprehensive R Archive Network" or CRAN is a collection of sites which collectively serve as the "official" repository for R material, R distributions, contributed extensions, and documentation for R.

cooperative automatic repeat request protocol, the prior adjacent node will retransmit the packet and the probability it will be successfully received is (1-p)p.[5] The probability that the packet is still not received is $(1- p)^2$ at which point a request for retransmission must be sent to the source node and the process repeated. The corresponding probability matrix for successful packet transmission is therefore:

$$P = \begin{bmatrix} (1-p)^2 & p(1-p) & p & 0 & 0 \\ 0 & (1-p)^2 & p(1-p) & p & 0 \\ 0 & 0 & (1-p)^2 & p(1-p) & p \\ 0 & 0 & 0 & p(1-p) & p \\ 0 & 0 & 0 & 0 & 1 \end{bmatrix} \quad (1)$$

The next step taken was to input the equation (1) transition matrix into the "markovchain" library in R. Note that the model developed in R provides a model for the Markov chain; this is not synonymous with a Markov decision process. The Markov chain represented in R furnishes a performance analysis tool that ideally will help inform or augment a Markov decision process study. The tool does not consider actions and rewards, as a Markov decision process does, and does not perform a stochastic optimization that identifies the best course of action given a particular set of objectives and constraints.

The use of a Markov chain to inform a Markov decision process in wireless sensor networks is not new. For example, Fei et al. [11] have modeled the process of moving sensor nodes as a Markov chain and the selection of moving direction being determined by a Markov decision process. In our case, a Markov chain will model the number of intermediate nodes and their associated transmission range R_t and a Markov decision process will be used to determine a minimum cost policy associated with the number of nodes and transmission ranges. To simplify the analysis, we will assume that the intermediate nodes are all alike in terms of capability and cost and that the transmission costs are proportional only to the transmission range R_t and not related to any other factor. The (simplified) Markov decision model we seek is one that minimizes the total weighted cost; this cost is dependent only upon the number of sensor nodes and their transmission range R_t.

To build a cost definition, a cost of λN is assigned to the sensor nodes present in the network where λ is the cost of the sensor and N is the number of sensor nodes. To associate a cost to the transmission range R_t, a cost of c(r) is assigned to the interval distance r separating each pair of successive sensors. The total transmission cost C_{trans} may then be defined as $C_{trans} = \sum_{i=1}^{N} c(r_i)$ where N is again the total number of sensor nodes and r_i is the interval distance cost over the i^{th} interval between sensors.

[5] The probability the packet will be forwarded by its previous node is the probability that the original packet was not originally received (1 – p) multiplied by the probability it was received correctly in at the prior node (p) thereby resulting in a probability of (1-p)p.

The association between the interval distance cost and sensor transmission power is made by assigning $c(r_i) = P_r$ where P_r represents the minimum power required to successfully transmit a data packet across the distance r from one sensor node to the next (i.e. to transmit over a hop). Note that in more sophisticated models c(r) would be the sum of several factors that takes into account path loss, a signal-to-noise ratio (SNR) constraint, and other factors [12]. Because we have previously imposed a simplifying requirement that the sensors be equidistant, $r_1 = r_2 = r_3 = r$ and, under a cooperative automatic repeat request assumption, R_t is required to be twice the distance r.[6] The total transmission cost can now be expressed as $C_{trans} = \sum_{i=1}^{N} c(r_i) = \sum_{i=1}^{N} 2 * (P_r) = 2NP_r$. If now let E_π represent the expectation operator, we seek a Markov decision model policy π that is defined by the following objective function:

$$\text{Minimize } E_\pi \lambda N + E_\pi C_{trans} = \text{Minimize } \lambda E_\pi N + 2E_\pi NP_r \quad (2)$$

IV. Solution Approach

The transition matrix developed in the previous section (equation 1) was input into R using the 3.5.0 version of R, R Studio, the "markovchain" library, and the "diagram" library which renders simple graphs based upon the transition matrix (only the 'plotmat' utility from this library was used). The 'markovchain" library was the primary analytical tool used.

After implementing the transition matrix in R, "markovchain" was tasked to return transition matrix values, identify transition and absorbing states, and to determine the fundamental matrix. In addition "diagram" was tasked to render a Markov state transition diagram.

Finally an attempt was made at using the Markov chain model to inform a Markov decision process that would optimize the linear combination of average sensor node power and the number of sensor nodes deployed. Several features of the "markovchain" library were explored for this purpose.

V. Results and Discussion

The Fig. 4 is a code snippet that details how the transition probability matrix was manifested into "markovchain" object.[7] The vector name "Transition" was assigned to the transition matrix and a test value of p = 0.8 was used to numerically evaluate the

[6] Recall that if the signal from the transmitting node (at a distance of $R_t = 2r$ units away) is not received, the preceding node (at a distance of r units away) will retransmit the packet. The $R_t = 2r$ requirement is not a universal feature of cooperative automatic repeat request protocols as there are many other ways to implement this protocol.

[7] The R script is included in Appendix 1 and an R Markdown document of the entire code and associated output is furnished in Appendix 2.

results. Variables a, b, and c were used as shorthand for $(1-p)^2$, p(1-p) and (1-p), respectively.

```
library(markovchain)
library(diagram)

# Set probability of sucessful transmission value
p <- 0.8

# Define variables in the transition probability matrix
a = (1-p)^2
b = p*(1-p)
c = 1-p

# Load transition matrix
nodes <- c("1","2","3","4","5")
Transition <- matrix(c(a,b,p,0,0,0,a,b,p,0,0,0,a,b,p,0,0,0,c,p,0,0,0,0,1),
                     nrow=5, byrow=TRUE)
row.names(Transition) <- nodes
colnames(Transition) <- nodes
Transition
```

Fig. 4. Code snippet for implementation of the transition matrix in R.

The next section of code, omitted here but included in both appendices, was to have provided a state transition diagram using the "diagram" library. Despite extensive testing with the "diagram" utility and the subroutine that was developed, the transition diagram displayed nonexistent links not represented in the transition matrix. Using the same subroutine on three-node networks produced accurate state diagrams but higher number of nodes proved to be too complex.

The next section of code, shown in Fig. 5, tells "markovchain" to identify transient and absorbing states. The results correctly indicated states 1, 2, 3, and 4 as being transient and state 5 as absorbing; this further indicated that the R-model was working correctly but did not otherwise contribute much additional insight.

```
# Identify transient and absorbing states
MarkChain <-new("markovchain",
                transitionMatrix = Transition,
                states = c("1","2","3","4","5"),
                name = "Markov Chain")
MarkChain

# Identify transient states
transientStates(MarkChain)


# Identify absorbing states
absorbingStates(MarkChain)
```

Fig. 5. Code snippet for function calls for transient/absorbing states in R.

The final section of code puts the Markov chain in canonical form and calculates the number of cycles until absorption. A Markov chain in canonical form has the properties illustrated in Fig. 6.

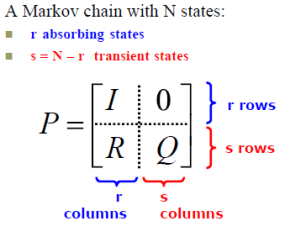

Fig. 6. Partition matrix [13]

Conversion was easily accomplished using the call "canonicForm(MarkChain)." The fundamental matrix E was also found by inverting E=(I – Q) producing an e_{ij} representing the expected number of times the process is in state j having started in transient state i.

The E matrix is also used to compute the expected number of transitions, starting at state i, until absorption. Fig. 7 illustrates depicts the code used.

```
# Put our Markov chain into canonical form
P <- canonicForm(MarkChain)
P

# Find Matrix Q
getRQ <- function(M,type="Q"){
  if(length(absorbingStates(M)) == 0) stop("Not Absorbing Matrix")
  tm <- M@transitionMatrix
  d <- diag(tm)
  m <- max(which(d == 1))
  n <- length(d)
  ifelse(type=="Q",
         A <- tm[(m+1):n,(m+1):n],
         A <- tm[(m+1):n,1:m])
  return(A)
}

Q <- getRQ(P)

# Find Fundamental Matrix
I <- diag(dim(Q)[2])
E <- solve(I - Q)
E


# Calculate time to absorption
c <- rep(1,dim(E)[2])
T_absorb <- E %*% c
T_absorb
```

Fig. 7. Code snippet for computing the canonical form of the transition matrix.

The result for algorithmically obtaining the canonical form is shown in Fig. 8 (the red partitions were added for clarity).

```
Markov Chain
 A  5 - dimensional discrete Markov Chain defined by the following states:
 5, 1, 2, 3, 4
 The transition matrix  (by rows)  is defined as follows:
     5    1    2    3    4
5 1.0 0.00 0.00 0.00 0.00
1 0.0 0.04 0.16 0.80 0.00
2 0.0 0.00 0.04 0.16 0.80
3 0.8 0.00 0.00 0.04 0.16
4 0.8 0.00 0.00 0.00 0.20
>
```

Fig. 8. Output snippet of transition probability matrix in canonical form.

The Q matrix was successfully extracted from the partitioned canonical matrix (see Fig. 9) and used to compute the fundamental matrix E (see Fig. 10).

```
> Q <- getRQ(P)
> Q
     1    2    3    4
1 0.04 0.16 0.80 0.00
2 0.00 0.04 0.16 0.80
3 0.00 0.00 0.04 0.16
4 0.00 0.00 0.00 0.20
>
```

Fig. 9. Output snippet of the Q matrix.

```
> E
          1         2         3         4
1 1.041667 0.1736111 0.8969907 0.3530093
2 0.000000 1.0416667 0.1736111 1.0763889
3 0.000000 0.0000000 1.0416667 0.2083333
4 0.000000 0.0000000 0.0000000 1.2500000
>
```

Fig. 10. Output snippet of the fundamental matrix.

By inspection of the top row of the fundamental matrix, it can be seen that states 1, 2, 3, and 4 are, on the average, transitioned 1.041667, 0.1736111, 0.8969907, and 0.3530093 times, respectively. We can assign the times t_1, t_2, t_3, and t_4, to the above state transitions to determine the total time taken to transmit a data packet from the source (node 1) to destination (node 5) as:

$$T_{total}=1.041667t_1+0.1736111t_2+0.8969907t_3+0.3530093\,t_4 \quad (3)$$

If we have $t_1 = t_2 = t_3 = t_4 = t$ then the above equation becomes:

$$T_{total}=1.041667t+0.1736111t+0.8969907t+0.3530093\,t \quad (4)$$

This corresponds to the absorption time computed by "markovchain" as can be seen by comparing the above result to the first element of the column vector in Fig. 11.

```
> T_absorb
      [,1]
1 2.465278
2 2.291667
3 1.250000
4 1.250000
>
```

Fig. 11. Output snippet for the absorption time computation.

Unfortunately none of the features examined provided informative guidance for a Markov decision process. Of moderate utility was the absorption time computation because the absorption time represented the total time the data packet is in the system; this can be viewed as either a cost or a constraint depending upon how the Markov decision process is formulated. Fig. 12 illustrates who absorption time changed as the transition probability p varied from 0.10 to 1.

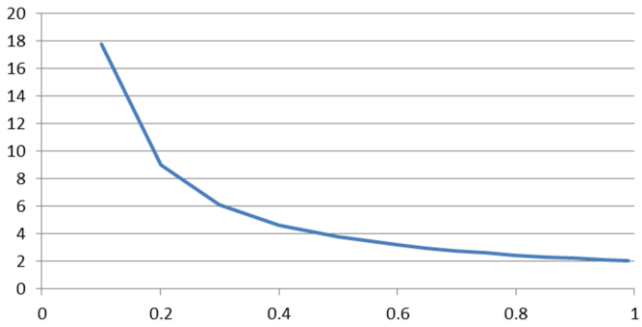

Fig. 12. Absorption time as a function of transition probability.

What was sought from this research was to formulate the objective function (Equation 2) as an infinite time Markov decision process with the optimal policy being derived from the Markov chain simulation of the optimal node number and transmission range r. This was initially believed to be an easy problem because the simplifying assumptions made it a one-dimensional problem (i.e. finding node placement and node separation on a line) versus a more complex two-dimensional problem of node placement on a lattice [9]. However there was no linkage between the first order "markovchain" library features to the parameters comprising Markov decision process.

The approach intended was to start with the standard method of defining the Markov decision process as a 4-tuple < S, A, P, R, T > where:

- S represents states, specifically as utilizing 1, 2, or 3 intermediate sensor nodes.
- A represents action, specifically positioning the sensor nodes so that they have the same separation distance.
- P represents the transition probability, specifically the probability of adding a sensor node after their separation distance is adjusted.
- R represents the reward, specifically the minimum cost obtained from the constrained objective function (Equation 2) after the node separation distance is adjusted.
- T represents the decision timeframe which for this case would have been infinite (note that system and component reliability considerations have been disregarded so the operating time is therefore assumed to be infinite).

The Markov chain was intended to model the number of intermediate nodes (states) which the Markov decision process would have optimized (i.e. a policy π^* would have been found that made an optimal selection of the number of sensor nodes). Since the node separation distance, transmission range R_t, and node transmitter power P_r were linked together and transformed into a cost (see Section III), it was also intended to model these parameters via a Markov chain and permit the Markov decision process to determine a minimum cost policy associated with the transmission range.

Crucially, the transition probability matrix developed by the Markov chain successfully modeled packet transition between the sensor nodes but did not represent the desired transition probabilities for the Markov decision process, namely the transition probability of adding a sensor node after node separation distance is adjusted. Modeling the packet transition was vital to this work but representing the transition probabilities tied to actions that are represented in the 4-tuple was also vital.

Available techniques for solution of infinite time Markov decision processes include the familiar methods of value iteration, policy improvement, and linear programming, along with approximation methods (for large Markov decision processes) and online learning (for problems where the transition probabilities are not initially known) [2].

VI. Conclusion and Future Research

An R-based Markov chain model for packet delivery in a multi-hop wireless sensor network was successfully demonstrated for a simplified 5-node mesh network model. Although it was not proven, it is believed that the model presented here is generalizable to more complex topologies. The Markov chain model, as implemented herein, was not useful in guiding the development of a Markov decision process. The principal reason for this was a failure to associate the first order "markovchain" library features to the parameters comprising the total cost Markov decision process. Future research, including further development of "markovchain" capabilities, may potentially correct this.

It should be noted that, for the sake of simplicity, many omissions were made that must be taken into account in future versions if the model is to ultimately reflect all the phenomenon present; these include data packet latency and delay issues, packet size,

bandwidth, data aggregation and routing complexities, quality of service (QoS) issues, overall geometry of the network, fault tolerance, and static network design assumptions that presume both the traffic and the wireless environment are static [14].

Broader consideration of available analytical tools is also warranted for use in future research. These include hidden Markov models [5], multi-agent Markov decision processes [2], and partially observable Markov decision processes [2], all of which furnish additional analytical power. Distributed computation of the optimum policy is an exciting direction for future research which allows the nodes to autonomously make real-time decisions based upon local information exchanges with other sensor nodes [14].

REFERENCES

[1] S. Vijayalakwhmi and S. Muruganand, *Wireless Sensor Networks: Architecture, Applications, Advancements*, Dulles, VA; Mercury Learning and Information, 2018

[2] M. Alsheikh, D. Hoang, D. Niyato, H. Tan, and S. Lin, “Markov Decision Processes with Applications in Wireless Sensor Networks: A Survey,” *IEEE Communications Surveys and Tutorials*, vol. 17, no. 3, pp. 1239-1267, April 2015

[3] K. Yu, L. Liu, J. Li, and H. Chen, “Mining Markov Blankets Without Causal Sufficiency,” *IEEE Transaction on Neural Networks and Learning Systems*, vol. 29, no. 12, pp. 6333-6347, December 2018.

[4] A. Erreygers, C. Rottondi, G. Verticale, and J. De Bock, “Imprecise Markov Models for Scalable and Robust Performance Evaluation of Flexi-Grid Spectrum Allocation Policies,” *IEEE Transactions on Communications*, vol. 66, No. 11, pp. 5401-5414, November 2018.

[5] H. Guo, W. Pedrycz, and X. Liu, “Hidden Markov Models Based Approaches to Long-Term Prediction for Granular Time Series,” *IEEE Transactions on Fuzzy Systems*, vol. 26, no. 5, pp. 2807-2817, October 2018.

[6] Q. Wang, G. Guo, L. Cao, and X. Xing, “Scheduling Strategy for Hidden Markov Model in Wireless Sensor Networks,” *Proceedings of the 34th Chinese Control Conference*, Hangzhou, China, July 28-30, 2015.

[7] Y. Rizk, M. Awad, and E. Tunstel, “Decision Making in Multiagent Systems: A Survey,” *IEEE Transactions on Cognitive and Developmental Systems*, vol. 10, no. 3, pp. 514-529, September 2018.

[8] W. Dargie and C. Poellabauer, *Fundamentals of Wireless Sensor Networks: Theory and Practice*, West Sussex, UK; John Wiley & Sons, 2010.

[9] V. Kumar, R. Patel, M. Singh, and R. Vaid, “Markov Model for Reliable Packet Delivery in Wireless Sensor Networks,” *International Journal of Computer Science Issues*, Vol. 8, Issue 3, No. 1, pp. 429 – 432, May 2011.

[10] G. Spedicato, T Kang, S. Yalamanchi, and D. Yadav, “The markovchain Package: A Package for Easily Handling Discrete Markov Chains in R.” https://cran.r-project.org/web/packages/markovchain/vignettes/an_introduction_to_markovchain_package.pdf . Retrieved April 19, 2019.

[11] X. Fei, A. Boukerche and R. Yu, "An efficient Markov decision process based mobile data gathering protocol for wireless sensor networks," *2011 IEEE Wireless Communications and Networking Conference*, Cancun, Mexico, pp. 1032-1037, 2011.

[12] A. Sinha, A. Chattopadhyay, K. Naveen, P. Mondal, M. Coupechoux, and A. Kumar, “Optimal Sequential Wireless Relay Placement on a Random Lattice Path,” *Ad Hoc Networks*, vol. 21, pp. 1-17, October 2014.

[13] F.S. Hillier and G.J. Lieberman. *Introduction to Operations Research*. 10th ed., McGraw-Hill, 2016

[14] Z. Lin and M. van der Schaar, “Automatic and Distributed Joint Routing and Power Control for Delay-Sensitive Applications in Multi-Hop Wireless Networks,” *IEEE Transactions on Wireless Communications*, vol. 10, no. 1, pp. 102113, January 2011