Robust and Adaptive Mobile Ad Hoc Routing with Attractor Selection

Narun Asvarujanon
Graduate School of Information Science and Technology, Osaka University, Japan
narun-a@ist.osaka-u.ac.jp

Kenji Leibnitz
National Institute of Information and Communications Technology (NICT), Japan
leibnitz@nict.go.jp

Naoki Wakamiya
Graduate School of Information Science and Technology, Osaka University, Japan
wakamiya@ist.osaka-u.ac.jp

Masayuki Murata
Graduate School of Information Science and Technology, Osaka University, Japan
murata@ist.osaka-u.ac.jp

ABSTRACT

In this paper, we propose a biologically-inspired routing protocol for mobile ad hoc networks. As biological systems are well-known for their self-adaptability in a changing environment, we adopt the mechanism of *attractor selection* in the next hop selection process. Our protocol, called *MARAS*, is a noise-driven and feedback-based on-demand routing protocol, which fully operates within the IEEE 802.11 protocol stack. The main contribution of this paper lies in the design of the route maintenance mechanism performed by using a feedback packet for each delivered data packet. Since MARAS is based on attractor selection, i.e., a biological mathematical model in the form of temporal differential equations, MARAS is inherently adaptive to dynamically changing environments. In addition, the route recovery can be achieved without using additional broadcast control packets like in most of the on-demand routing protocols, e.g., AODV. In comparative evaluation, MARAS can achieve higher delivery efficiency while having lower overhead than AODV in the failure scenarios. In mobility scenarios, MARAS and AODV achieve roughly the same performance. We also compare MARAS to another biologically-inspired protocol, AntHocNet, and observe much lower overhead.

Categories and Subject Descriptors

C.2.2 [**Computer-Communication Networks**]: Network Protocols—*Routing protocols*

General Terms

Design, Performance, Algorithms

Keywords

Ad hoc networks, routing protocol, biologically-inspired networking, attractor selection mechanism

1. INTRODUCTION

Mobile Ad Hoc Networks (MANETs) have been receiving a lot of research attention in the last few decades. MANETs differ from traditional wired networks because they are independent of a fixed infrastructure; this allows the mobile nodes to move freely and makes many useful applications possible, e.g., rescue mission in an infrastructure-less area. However, this flexibility comes at the cost of difficulties in routing. Examples of these difficulties are dynamic topology changes, limited bandwidth and energy, and incomplete network information due to limited transmission range.

Many MANET routing protocols have been proposed in the past and they can be distinguished into three main categories: *proactive*, *reactive (on-demand)*, and *hybrid* protocols. We focus our research on on-demand protocols as it is crucial to preserve wireless network resources and utilize them only when required. In this aspect, proactive protocols are not appropriate because of the periodic routing information exchange, like in DSDV [12]. Among the on-demand protocols, AODV [11] is more scalable and more adaptive than DSR [7] as shown in [13]. However, AODV has its own weakness, which is high control overhead caused by flooding. Therefore, we aim at designing a new routing protocol, which is more adaptive under unstable conditions in the network and causes lower overhead than AODV.

In our protocol, we consider a biologically-inspired mechanism because biological systems are well-known for their robustness and adaptability. There is a lot of research adopting mechanisms inspired by biology, e.g., ant colony optimization (ACO) [4] which is based on swarm intelligence [2], neural network which is inspired by the human brain, and genetic algorithm which is inspired by the evolutionary biology. For MANETs, many biologically-inspired routing protocols have been proposed and most of them are based on swarm intelligence, e.g., AntHocNet [3] and HOPNET [15]. Note that our protocol however uses a mechanism from cell biology called *attractor selection* [8] and is not based on swarm intelligence. The reason is that the concept of routing with the attractor selection mechanism has been found simple and robust to failures [9]. Also, we attempt to discover a new alternative other than social insects-based swarm intelligence.

Our adaptive mobile ad hoc routing with attractor selection (MARAS) is an extended work from [10]. Unlike the original protocol, which focused only on the basic routing mechanism using simplified assumptions on packet level, we perform a more in-depth evaluation of MARAS in the IEEE 802.11 protocol stack. MARAS is a noise-driven on-demand protocol which uses feedback of delivered data packets from the destination for route maintenance. Using the feedback information along with the attractor selection mechanism allows MARAS to recover from link failures without issuing any additional broadcast control message like AODV. Evaluation results in this paper show that MARAS has higher delivery efficiency and lower transmission overhead than AODV in failure scenarios, while keeping a similar performance to AODV in mobility scenarios. We also compare MARAS to another biologically-inspired protocol, AntHocNet. Except in scenarios with low failure occurrences and low traffic load, MARAS generally has much higher delivery efficiency and much lower overhead than AntHocNet and in the rest of considered scenarios, which shows that MARAS is more adaptive.

The rest of this paper is organized as follows. First, we introduce the attractor selection mechanism and the derived mathematical model in Section 2. Next, we describe our protocol in Section 3. In Section 4, the evaluation results are presented and discussed. Finally, we conclude this paper and list future work in Section 5.

2. MATHEMATICAL MODEL

In this section, we introduce the background of the attractor selection mechanism and our derived mathematical model.

2.1 Attractor Selection Mechanism

The attractor selection mechanism is modeled after the behavior of *E. coli* cells, which is capable of adapting to dynamically changing nutrient conditions in the environment without an embedded rule-based mechanism [8]. A mutant *E. coli* cell has a metabolic network consisting of two mutually inhibitory sequences of chemical reactions which synthesize two corresponding nutrients. When one of the nutrients becomes scarce, the protein concentration activating a sequence for the missing nutrient increases to return the cell to a stable condition. However, there is no explicit rule-based mechanism to switch between the sequences of chemical reactions. In [8], a model describing this bistable behavior of protein concentrations m_1 and m_2 is proposed as

$$\frac{dm_1}{dt} = \frac{s(\alpha)}{1 + m_2^2} - d(\alpha)m_1 + \eta_1 \tag{1}$$

and

$$\frac{dm_2}{dt} = \frac{s(\alpha)}{1 + m_1^2} - d(\alpha)m_2 + \eta_2, \tag{2}$$

where $s(\alpha)$ and $d(\alpha)$ are the rate coefficients of protein synthesis and degradation, respectively. Both of them depend on α which represents the cell activity or cell volume growth. The η_1 and η_2 are independent white noise in gene expression.

Each state in the metabolic network is represented by a pair of protein concentrations (m_1, m_2). The equilibrium conditions in the metabolic network are called *attractors*. When the cell becomes unstable due to external influences or internal noise, its gene expression state will be driven to other attractors to return the cell to a stable condition. As there are more than one possible stable conditions, this mechanism selects a suitable attractor among multiple attractors, which is called attractor selection.

We extend the model from two alternatives to M alternatives based on Eqns. (1) and (2). Let m_i be the value representing whether the i^{th} choice should be selected. Moreover, let us define the M-dimensional vector $\vec{m} = (m_1, \ldots, m_M)$. The attractor selection among M alternatives shall have the general form as

$$\frac{d\vec{m}}{dt} = f(\vec{m}) \times \alpha + \vec{\eta}, \tag{3}$$

where α expresses the goodness of the current condition and $\vec{\eta} = (\eta_1, \ldots, \eta_M)$ is the vector of the noise affecting the selection.

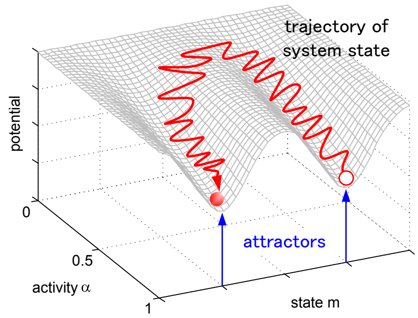


Figure 1: Behavior of attractor selection system

The activity α is the main parameter which controls the influence of randomness on attractor selection. When the current condition of the system becomes undesirable, the activity decreases. As a result, the value of term $f(\vec{m}) \times \alpha$ will decrease and a larger effect from noise $\vec{\eta}$ will take place to shift the system to another attractor by a random walk. Once the system reaches a suitable attractor, the activity will increase and the effect of noise will be suppressed, which then allows the system to become stable again.

In Figure 1, we show the general principle of the attractor selection concept. The x-axis shows a one-dimensional state m, where some possible values of m are the attractors, the y-axis is the activity α, and the z-axis indicates the energy potential defined by $f(m) \times \alpha$. The current system state is illustrated as a circle which is constantly in motion due to the effect of the noise. It can be observed that when the activity is high, changing the system's state is difficult because of the steepness of the potential landscape. On the other hand, when the activity is low, the landscape becomes smoother and changing the state can be achieved easier by only the small effect of noise.

At first, the concept of having noise in the system may look undesirable. However, adding noise into the system makes it in general more robust to external fluctuations. In sensor networks, noise and random walk can provide load-balancing and scalable properties [1]. Moreover, getting stuck in local minima can be avoided using noise and random walk as explained in [14].

2.2 Application to MANET Routing Protocol

The attractor selection is adopted in our protocol for next hop selection among neighbors. Hence, we map the vector of neighbors to $\vec{m}$, which contains *state value* m_i, indicating whether the i^{th} neighbor should be selected among M neighbors. We further map activity α to the information reflecting the goodness of the current routing condition. Since the next hop selection shall provide a single next hop neighbor as the solution, we design the controlling function of attractor selection as follows. For all neighbors i:

$$\frac{dm_i}{dt} = \frac{s(\alpha)}{1 + m_{max}^2 - m_i^2} - d(\alpha)m_i + (1 - \alpha) \times \eta_i, \tag{4}$$

where $m_{max} = \max_{j=1,\ldots,M}(m_j)$, $s(\alpha) = \alpha[\beta\alpha^\gamma + \varphi^*]$, $d(\alpha) = \alpha$, $\varphi^* = 1/\sqrt{2}$, and η_i is the white noise with mean of 0 and variance of 1. Parameters β and γ control the influence of activity over state values and we use empirical values $\beta = 10$ and $\gamma = 3$ throughout this study. The term $(1-\alpha)$ additionally suppresses the effect of noise when the activity is high.

In case of high activity α, Eqn. (4) gives the $\vec{m}$, which has a single high m_i value and $M-1$ low values. Then the deterministic solution can be obtained by selecting the maximum value. While

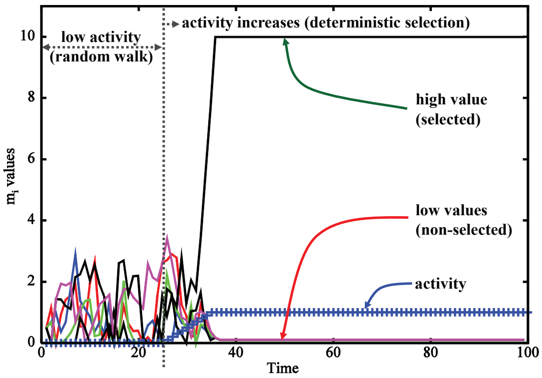


Figure 2: Dynamics of M values from attractor selection model

in case of low activity α, Eqn. (4) gives a random $\vec{m}$ where each member m_i has roughly the same value. This gives another state value, which was previously not selected but has now become more suitable for the current condition, a chance to become the new maximum value easily requiring only small effect of noise. According to this approach, the appropriate selection can be made.

The dynamics of $M = 6$ alternative values from Eqn. (4) is shown in Figure 2 where the solid lines represent the m_i values while the '+' line represents the activity α. From time $t = 0$ to 25, α is low, therefore, each value m_i receives more effect from noise and has a random value. After a solution is found at time $t = 25$, α starts increasing. As a result, the gap between selected value and non-selected values increases as well and becomes stable with one high value and $M - 1$ low values once α is equal to 1.0, which indicates that the system reaches a suitable attractor.

3. OUR ROUTING PROTOCOL

In this secion, we explain the details of our protocol, Mobile Ad hoc Routing with Attractor Selection (MARAS). MARAS uses the feedback packets to evaluate the path that the data packets have taken and notifies the path condition to each node along the path. Therefore, we assume a bidirectional connectivity between each neighbors. However, MARAS can also operate in a network containing unidirectional links, as explained later in Section 3.4.3.

3.1 Route Establishment

We adopt the broadcast route discovery mechanism from AODV and make a few modifications. In our protocol, when a node has data to send but no route is available, a *route-request packet* (RREQ) is broadcast from the source node and re-broadcast by other nodes until it reaches the destination. Each RREQ packet has a unique ID, which is used to detect and drop a duplicated RREQ packet. The previous hop of a valid RREQ packet is memorized for sending the route reply packet back to the source in the future.

When the RREQ packet arrives at the destination, a *route-reply packet* (RREP) is generated and is forwarded in unicast manner via the memorized reverse path to the source. When a node on the path receives the RREP packet, it sets up the route entry for the source of RREP, i.e., the destination of data packet. This route entry favors the selection of the previous hop of the RREP packet. Moreover, the route entry is marked with a maximum activity $\alpha = 1$ because of the availability of a route to the destination. Once the RREP arrives at the source node, the data packet forwarding begins.

Due to the random selection of the next hop in the low activity case, a node which was not on the pre-established path occasionally receives a data packet. If the current node has no route entry for that destination, then it will set up a new random vector which contains equal state values m_i of $\lambda = 0.5$ for every neighbor i and starts routing by the effect of noise. The random walk phase continues until a feedback is received and the attractor selection mechanism is applied to the routing information of the current node.

3.2 Routing Information

The routing information stored at each node in the route entry are as follows.

- *Destination address* is used for looking up the corresponding route entry when a data packet is received.
- *Neighbor vector* $\vec{n} = (n_1, n_2, \ldots, n_M)$ is a list of neighbor addresses, maintained by HELLO packets like in AODV.
- *Attractor selection vector*, called *routing vector* $\vec{m} = (m_1, m_2, \ldots, m_M)$ has the same dimension as the neighbor vector and contains the *state values*, where each state value is mapped to a neighbor address in the neighbor vector. These state values are used to determine the next hop of each data packet.
- *Activity* α reflects the current goodness condition of the path to the destination. The routing vector is updated according to this value, allowing the next hop selection to adapt to the current condition.
- *Precursor list* contains pairs of the address of the source node and the address of the most recent neighbor that forwarded the data packet originating at that source node to the destination via the current node.
- *Feedback window* is a sliding window where each frame contains the travelled hop count of the feedback packet, which originates at the destination and is sent via the current node. Each frame is added to the feedback window on the reception of a feedback packet and will be deleted after *window interval* $T = 1.0\,$s to avoid using outdated information.

3.3 Data Packet Forwarding

Next hop selection in data packet forwarding is controlled by the attractor selection mechanism. Using attractor selection, MARAS selects the neighbor which has the *maximum state value* in the routing vector as a next hop because the maximum value shows the highest chance of that neighbor on delivering the data packet to the destination. The data packet is forwarded to this next hop and the process repeats itself until it reaches the destination.

The concept of attractor selection along with the maximum state value favors the next hop selection in a way that MARAS will keep selecting the same next hop as long as the activity is high. When the activity drastically decreases, which reflects an undesirable condition, state values first become closer to one another. Then, the effect of noise changes the state values and allows a different candidate to be chosen. Hence, MARAS is able to recover from such conditions without using additional explicit control messages.

3.4 Route Maintenance

MARAS maintains the routes as long as they are being used and removes unused routes after a certain period of time to save the memory resource required to maintain them. In order to keep the routing information up-to-date, MARAS uses the feedback packet to learn the current condition of the network. The route maintenance mechanisms are explained in this section.

3.4.1 Feedback Packet

Upon the data packet arrival at the destination, a feedback packet is generated and sent back to the source. The feedback packet utilizes the memorized previous hop in the precursor list at each intermediate node to take the most recent route back to the source and to avoid getting lost. During its journey, its travelled hop count information is left in the *feedback window* of each intermediate node for the purpose of activity calculation. The hop count information in the feedback window is deleted after *window interval* T to avoid using outdated information.

3.4.2 Activity Calculation

The activity of each routing vector is calculated upon the arrival of a feedback packet at time t_0 based on the travelled hop count of the most recent feedback packet and the minimum travelled hop count in the feedback window:

$$\alpha(t_0) = \frac{\min_{t_0 - T < t \leq t_0} w(t)}{w(t_0)}, \tag{5}$$

where $w(t)$ is the travelled hop count of the feedback packet which arrives at time t. Until the next arrival of the feedback packet, $\alpha(t_x) = \alpha(t_0)$, where $t_x \geq t_0$.

This activity changes according to the hop count to the destination in the range between 0 and 1. If the hop count to the destination becomes larger, then it means that the current path to the destination is no longer appropriate and an attempt to find a better path should be made. Therefore, the activity will decrease and the effect from noise will induce a random walk. On the other hand, once a shorter path is found, $\alpha(t)$ will immediately become 1, and MARAS will keep using this path until another change occurs in the network.

3.4.3 Activity Decay and Routing Vector Update

When the route is broken, data packets cannot arrive at the destination and there will be no feedback packet returning to the source. In such condition, the activity must decay to let the system escape from a stalling condition. In case of unidirectional links, the absence of feedback is treated similarly to the case of a broken link. This implies that paths consisting of bidirectional links are preferably chosen.

In our protocol, given the current time t and the most recent feedback packet arrival at t_0, we use the simple activity decay equation on the stored activity:

$$\alpha(t) = \begin{cases} \alpha(t_0) - \delta & \text{if } t - \tau \leq t_0 < t \\ \alpha(t - \tau) - \delta & \text{otherwise,} \end{cases} \tag{6}$$

where the decay constant $\delta = 0.1$ is used for the current implementation. The decay process is periodically performed over interval $\tau = 1.0\,\text{s}$. The activity decay mechanism is performed regardless of the feedback packet arrival. Therefore, when there is no incoming feedback packet, the activity will continuously decay and the routing vector will be updated using the decayed activity.

To keep the information in the routing vector consistent with the value of activity, the routing vector is always updated after there is any change of the activity value.

3.4.4 Attractor Selection-based Route Recovery

In MARAS, data packets occasionally take a random walk looking for a new path to the destination because of the noise term and the random routing vector as explained in Section 3.1. This behavior of data packets inherently contributes to route recovery. As such, MARAS does not have any mechanism designated for route recovery.

However, using the data packet as a route recovery packet has a drawback of a possibly lower delivery efficiency due to loss of data packets. Since the delivery efficiency is crucial in most of the communication, the random walk packet should travel as many hops as possible to achieve the high delivery efficiency. On the other hand, the longer path of the random walk packet also causes larger overhead and interference. This trade-off is considered in our implementation and we introduce the *random walk range* ρ and *random walk threshold* θ for this purpose. Whenever the activity is lower than the random walk threshold θ, the maximum TTL is limited to random walk range ρ instead of the default value to avoid the negative effect of a long path. We use empirical values $\rho = 10$ and $\theta = 0.6$ throughout this study.

4. EVALUATION

We evaluate MARAS by performing simulations with the network simulator QualNet. MARAS is compared to AntHocNet and AODV. We use the code of AntHocNet from the developers available at [5] and the implementation of AODV in QualNet 4.0. Three different variants of AODV, which are the standard AODV, the standard AODV with local route repair feature (*AODV+L*), and the standard AODV with both local route repair and allowing RREP by intermediate node (*AODV+LI*), are used in this evaluation.

We consider two metrics in this evaluation: delivery efficiency and transmission overhead. The *delivery efficiency* is the ratio of the number of successfully delivered data packets at the destination to the number of data packets sent from the source. The *transmission overhead* is the ratio of the sum of all unicast and broadcast transmissions in the network for the whole simulation to the number of the successfully delivered data packets. This metric reflects the amount of network load inflicted by the delivery of each data packet.

4.1 Simulation Configurations

The evaluation is separated into two main scenarios: a failure scenario and a mobility scenario. In this section, we describe simulation configurations for both scenarios.

The area of the evaluation scenario is $1500 \times 1500\,\text{m}^2$ for both scenarios. Using the node placement tools available in Qualnet, nodes are placed uniformly in the failure scenario and randomly in the mobility scenario. The number of nodes is varied from 121, 169, to 256 in the same area but only the results from the 256 nodes case are discussed here due to the space limitations.

Each node in the simulation uses the IEEE 802.11b wireless module with data rate of 2 Mbps which is the common configuration in many protocol evaluations [3, 15]. The approximate radio range is 510 m as we use the free-space model of QualNet without propagation fading. Additionally, we assume an infinite wireless interface buffer at each node. Regarding the traffic, CBR is used with UDP as a transport layer protocol to avoid effects from the congestion control mechanisms of TCP. We use CBR of 8 kbps which sends out 10 packets per second. The results which are discussed in this section are the average values of each traffic session from 100 simulation runs. In both scenarios, the simulation time is 1000 s where the traffic generation starts and ends at the same time as the simulation.

The specific parameters of MARAS are summarized in Table 1. The parameters of AntHocNet are set according to the sample configuration file provided with the code in [5]. The other parameters of AODV and MARAS, which are not given here, are default values according to the implementations in QualNet 4.0.

In the failure scenarios, a failure model is used to simulate topology changes which are caused by joining and leaving nodes. We force approximately 25% of all nodes to fail at the same time by switching their wireless interfaces off using the API available in QualNet. Consequently, link failures occur and the route recovery performance can be evaluated. Failing nodes are randomly selected among all nodes excluding the sources and the destinations. The

Table 1: Simulation parameters of MARAS

Category	Parameter Name	Value
Attractor selection	High value β	10
	Activity exponent γ	3
Activity calculation	Window interval T	1.0 s
	Decay constant δ	0.1
	Decay interval τ	1.0 s
Routing	Random vector's initial value λ	0.5
	Random walk threshold θ	0.6
	Random walk range ρ	10

configurations of the number of failures are between 0 and 90 with the incremental step of 10 occurrences. The failure interval is calculated by dividing the total simulation time by the number of failure occurrences, which ranges between 100 s in case of 10 failures to 11.11 s in case of 90 failures. We use this variable interval to maintain the same number of active nodes and to evaluate the effect of the failure frequency. The first group of nodes is forced to fail at 0 s. Iteratively after the failure interval, the previously failing nodes recover from failures and a new group of randomly selected nodes fails. Note that the value 0 means no failure occurrences.

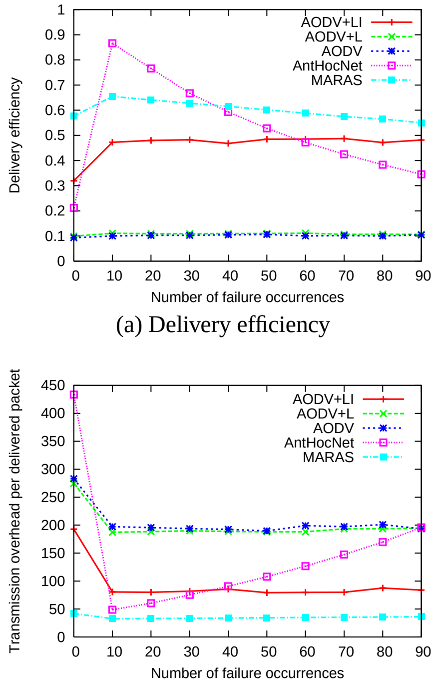
For the mobility scenarios, the random waypoint mobility model (RWP) is used as it is the most common mobility model in evaluations. In RWP, there are three main parameters: maximum speed, minimum speed, and pause time. A mobile node under this model will select a random target coordinate and a random speed within the range of minimum and maximum speed for moving toward the target. Once the mobile node reaches the target, it will remain still at that target for *pause time* period before repeating the process again. In our mobility scenario, we use 0 as a minimum speed and 2, 5, and 10 m/s are used for the maximum speed. The pause time is set as 0 to study the pure effect from mobility.

4.2 Failure Scenario

In this section, the evaluation results from the failure scenario are shown and discussed. This scenario has two source/destination pairs. The sources are the two nodes closest to the lower left corner of the simulation area and the destinations are the two nodes closest to the upper right corner. The purpose of the selection of source and destination pairs is to cause interference between both sessions. The simulation results using only a single traffic session showed nearly similar results and are omitted here due to space limitations.

In Figure 3, MARAS achieves higher delivery efficiency and lower transmission overhead per successfully delivered packet than AODV for all cases. Comparing the results between the two traffic sessions and the single traffic session, the delivery efficiency of MARAS drops only approximately 5–10%, but the delivery efficiency of AODV drops 10–15%, which shows that MARAS can handle a higher amount of traffic than AODV before the performance degrades. Regarding AntHocNet, even though the delivery efficiency is higher for less dynamic conditions with low failure occurrences, it becomes much lower and drops faster over the increasing number of failure occurrences. Moreover, the overhead of AntHocNet increases much faster than AODV+LI and MARAS over the increasing number of failure occurrences. Therefore, it is sufficient to say that MARAS is more robust and adaptive against large number of failures and changes in topology than AODV and AntHocNet.

Note that the sudden changes between 0 and 10 on the x-axis of every graph are caused by the different number of active nodes.



(a) Delivery efficiency

(b) Transmission overhead

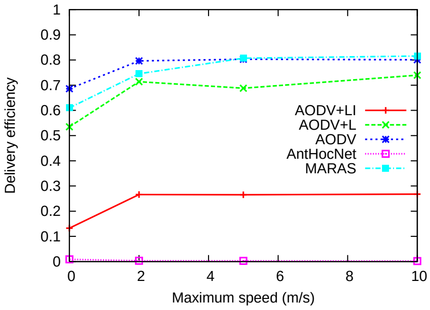
Figure 3: Evaluation results against number of failure occurrences

As the failure model puts 25% of nodes into inactive state, there are less HELLO packets, less radio interference, and less effects from broadcast storm. Therefore, higher delivery efficiency, lower overhead, and shorter paths are expected in such situation.

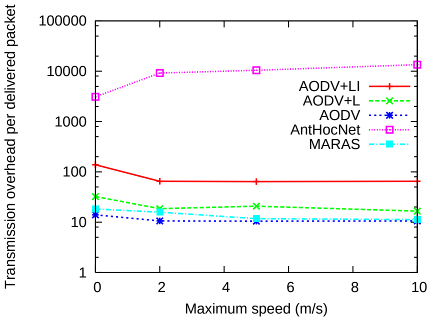
MARAS performs better than AODV and AntHocNet because of less usage of broadcast control packets. In case of AODV, reactive route recovery in broadcast manner is always used. Therefore, the overhead of AODV is very high because of the effect of broadcast storm. Since AODV+LI uses less route error broadcast packets, its overhead is lower than the other two. Moreover, AODV also suffers from the loss of HELLO packets in congested channels which causes even higher overhead. AntHocNet can avoid using only broadcast control packets upon error by gradually changing the routing information using a proactive approach. Therefore, the amount of overhead in AntHocNet grows when the network dynamic increases because the proactive approach cannot keep up with the changes and the broadcast route repair needs to be performed. On the other hand, MARAS has no explicit route recovery mechanism and uses only the feedback packet per each delivered data packet. As the number of feedback packets is in proportion to the traffic, not to the network dynamic, it can be observed that MARAS also has an almost constant amount of overhead. Moreover, lower overhead is achieved by lower interference of unicast feedback packets and the robustness and adaptability of the attractor selection mechanism.

4.3 Mobility Scenario

In this section, the evaluation results from the mobility scenario are shown and discussed. In this evaluation, 10 concurrent traffic sessions were performed. The source and the destination pairs are selected randomly among 256 nodes. However, each node can be either the source or the destination for only one traffic session to avoid the bottleneck problem.



(a) Delivery efficiency



(b) Transmission overhead

Figure 4: Evaluation results against maximum speed

The delivery efficiency results are shown in Figure 4(a). First, the delivery efficiency of AntHocNet totally deteriorates in this high traffic load and high network dynamic scenario due to its proportionally increasing overhead as seen in failure scenarios. Next, the delivery efficiency of AODV and AODV+L become much better than in the cases of the failure scenarios because of a better chance to reach the destination node directly due to mobility. However, the delivery efficiency of AODV+LI becomes worse because AODV+LI relies too much on the stored routing information, which is likely to be outdated. Comparing MARAS to AODV, the delivery efficiency of AODV is slightly larger in the scenarios where the maximum speed is less than 5 m/s, but MARAS has slightly higher delivery efficiency when the maximum speed becomes larger. However, these differences in terms of both delivery efficiency and overhead between MARAS and AODV are not significant and it can be concluded that MARAS has approximately the same level of performance as AODV in this mobility scenario.

5. CONCLUSION AND FUTURE WORK

In this paper, we presented MARAS, an adaptive routing protocol for MANETs inspired by the attractor selection mechanism. This mechanism is formulated by nonlinear stochastic differential equations with a control factor, called activity, which controls the influence of randomness in the selection process. Feedback packets are used to evaluate the route that the data packet takes and to update the activity at each node in the route by using the hop count information, allowing the route to react to changes in the network without creating additional control overhead on changes. As a result, MARAS achieves a higher delivery efficiency and a lower overhead than the other well-studied routing protocols, AODV and AntHocNet, because of its adaptability to frequent failures.

According to the evaluation results, AntHocNet with fine-tuned parameters from [6] does not perform well in our simulation settings. AODV performs quite well in both scenarios but requires the manual selection of optional features to adapt to a new condition. However, MARAS is capable of maintaining its performance throughout all the evaluated scenarios in this study without the need of tuning the parameters to adapt to new conditions as in other protocols. Hence, it can be concluded that MARAS is suitable as an adaptive general purpose routing protocol for MANETs. Although not shown in the paper, we also conducted scalability tests and verified that MARAS is more scalable than AODV against the increased node density.

Currently, the parameters used in the evaluation are merely empirical values. As future work, we would like to investigate the effects of parameters of MARAS, i.e., ρ, β, and γ to further improve the performance of MARAS for specific applications or environments in addition to its general purpose adaptability.

6. ACKNOWLEDGMENT

This research was partially supported by the “Global COE (Centers of Excellence) Program” of MEXT, Japan and the ‘Early-concept Grants for Exploratory Research on New-generation Network” of NICT and MEXT, Japan, Grant-in-Aid for Scientific Research (B) 22300023

7. REFERENCES

[1] C. Avin and C. Brito. Efficient and robust query processing in dynamic environments using random walk techniques. In *Proceedings of the 3rd International Symposium on Information Processing in Sensor Networks (IPSN)*, pages 277–286, Apr. 2004.

[2] E. Bonabeau, M. Dorigo, and G. Theraulaz. *Swarm Intelligence: From Natural to Artificial Systems*. Oxford University Press, 1999.

[3] G. Di Caro, F. Ducatelle, and L. Gambardella. AntHocNet: An adaptive nature-inspired algorithm for routing in mobile ad hoc networks. *European Transactions on Telecommunications*, 16(5):443–455, Sept. 2005.

[4] M. Dorigo, G. Di Caro, and L. M. Gambardella. Ant algorithms for discrete optimization. *Artificial Life*, 5(2):137–172, Apr. 1999.

[5] F. Ducatelle. AntHocNet4.zip: code for the use of AntHocNet in the QualNet 4.0 network simulator. Available at `http://www.idsia.ch/~frederick/anthocnet/anthocnet.html`.

[6] F. Ducatelle. *Adaptive Routing in Ad Hoc Wireless Multi-hop Networks*. PhD thesis, Università della Svizzera italiana, May 2007.

[7] D. B. Johnson, D. A. Maltz, and J. Broch. DSR: the dynamic source routing protocol for multihop wireless ad hoc networks. *Ad Hoc Networking*, pages 139–172, Jan. 2001.

[8] A. Kashiwagi, I. Urabe, K. Kaneko, and T. Yomo. Adaptive response of a gene network to environmental changes by fitness-induced attractor selection. *PLoS ONE*, 1(1):e49(1–10), Dec. 2006.

[9] K. Leibnitz, N. Wakamiya, and M. Murata. Self-adaptive ad-hoc/sensor network routing with attractor-selection. In *Proceedings of the IEEE Global Telecommunications Conference (GLOBECOM)*, pages 1–5, Nov. 2006.

[10] K. Leibnitz, N. Wakamiya, and M. Murata. A bio-inspired robust routing protocol for mobile ad hoc networks. In *Proceedings of the 16th International Conference on Computer Communications and Networks (ICCCN)*, pages 321–326, Aug. 2007.

[11] C. Perkins, E. Belding-Royer, and S. Das. Ad hoc on-demand distance vector (AODV) routing. RFC 3561, July 2003.

[12] C. Perkins and P. Bhagwat. Highly dynamic destination-sequenced distance-vector routing (DSDV) for mobile computers. In *Proceedings of the Conference on Communications Architectures, Protocols and Applications (SIGCOMM)*, pages 234–244, Aug. 1994.

[13] C. Perkins, E. Royer, S. Das, and M. Marina. Performance comparison of two on-demand routing protocols for ad hoc networks. *IEEE Personal Communications*, 8(1):16–28, Feb. 2001.

[14] B. Selman, H. Kautz, and B. Cohen. Noise strategies for improving local search. In *Proceedings of the National Conference on Artificial Intelligence (AAAI)*, pages 337–343, July 1994.

[15] J. Wang, E. Osagie, P. Thulasiraman, and R. K. Thulasiram. HOPNET: A hybrid ant colony optimization routing algorithm for mobile ad hoc network. *Ad Hoc Networks*, 7(4):690–705, June 2009.