



In order to provide any QoS guarantees in MANETs, we must make certain assumptions concerning the network dynamics. At first, since nodes only have limited transmission radius, a minimal node density must exist that connectivity among the nodes can be made and the destination node can be reached by some route from the source node. The higher the mobility of the nodes is, the more difficult it is to maintain connectivity. This is defined in [11] that the network should be *combinatorially stable*, i.e., the changes in topology occur sufficiently slowly to allow successful propagation of all topology updates as necessary.

Mohapatra *et al.* [1] discuss important issues of QoS in MANETs and describe methods on enhancing QoS at different layers of the protocol stack, for example, by using improved channel techniques on physical layer or by using IEEE 802.11 DCF mode to overcome the hidden terminal problem. However, we will focus in the following on methods operating on network layer. There are several proposals for QoS support, e.g. [12], [13], [5], [14], which are built on methods for estimating and reserving bandwidth or using a combination of several metrics [15]. In our approach we will focus on the packet delivery ratio as main metric which provides an indication of how well the destination node can be reached in the presence of topology changes. The idea is to use a simple decision scheme for determining the next hop which can quickly and autonomously overcome sudden connectivity failures in the network.

### III. ATTRACTOR SELECTION MODEL INSPIRED FROM CELL BIOLOGY

Our proposed routing method is based on each network node applying *adaptive response by attractor selection* (ARAS) to select its next hop. ARAS is a biologically inspired method for adaptively selecting one among several candidates which best reflects the current situation in a dynamic environment. ARAS is originally a model for its host *E. coli* cells to adapt to changes in the availability of a nutrient for which no molecular machinery is available for signal transduction from the environment to the DNA [2].

Two key principles are used in ARAS. The first is the concept of attractors to describe the multiple states of gene expression. An *attractor* is the region to which the orbit of a dynamic system recurrently returns regardless of the initial conditions [16]. Even if a state is perturbed by fluctuations, the system state will be drawn over time to an attractor. Therefore, the second principle which is utilized in ARAS is to introduce a small *inherent noise term* to the system which permits adaptation to new states and increases the robustness of the system to externally introduced fluctuations. In nature, noise always exists and no process or entity is purely deterministic. For example, if we consider cells of the same type, the values of the quantities describing them will vary from cell to cell and for a single cell, these values will also fluctuate over time [16].

Basically, we can outline the attractor selection method as

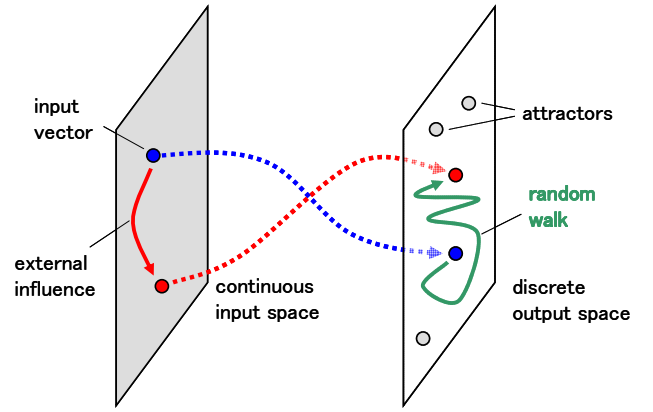


Fig. 2. General concept of ARAS

follows. Using a set of differential equations of the form

$$\frac{dm_i}{dt} = f(m_1, \dots, m_M) \times \alpha + \eta_i \quad i = 1, \dots, M$$

we describe the dynamics of an  $M$ -dimensional system. Each differential equation has a stochastic influence from an inherent Gaussian noise term. Additionally, we introduce an *activity*  $\alpha \in [0, 1]$  which changes the influences from the noise terms. For example, if  $\alpha$  is large, the system behaves rather deterministic and converges to attractor states defined by the structure of the differential equations. However, for small  $\alpha$  the noise term dominates the behavior of the system and essentially a random walk is performed. When the input values (*nutrients*) require the system to react to the modified environment conditions, activity  $\alpha$  changes accordingly causing the system to search for a more suitable state, see Fig. 2. This may involve that  $\alpha$  causes a previously stable attractor to become unstable.

#### A. Mathematical Model

Consider a set of  $M$  alternatives among which one will be selected. Let  $m_i$  be the proportion of selecting  $i$  as  $m_i$  and

$$\mathbf{m} = [m_1, \dots, m_M]^T$$

as the vector over all  $m_i$ . The dynamic behavior of each  $m_i$  is characterized by the stochastic differential equation system given in Eqn. (1).

$$\frac{d\mathbf{m}}{dt} = \frac{s(\alpha)}{1 + \max(\mathbf{m})^2 - \mathbf{m}^2} - d(\alpha)\mathbf{m} + \boldsymbol{\eta} \quad (1)$$

The functions  $s(\alpha)$  and  $d(\alpha)$  are the rate coefficients of mRNA synthesis and degradation in the original biological model, respectively. They are both functions of  $\alpha$ , which represents cell activity or vigor.

$$s(\alpha) = \alpha [\beta \alpha^\gamma + \varphi^*] \quad (2)$$

$$d(\alpha) = \alpha \quad (3)$$

The parameters  $\beta$  and  $\gamma$  in Eqn. (2) are factors which influence the mapping of activity to the output probabilities and we use  $\beta = 50$  and  $\gamma = 3$  throughout this study. The constant  $\varphi^*$  is a

special offset point which we will discuss below. For the sake of simplicity we also define the effective growth rate  $\varphi$ .

$$\varphi(\alpha) = \frac{s(\alpha)}{d(\alpha)} \quad (4)$$

Furthermore, the vector given by

$$\boldsymbol{\eta} = [\eta_1, \dots, \eta_M]^T$$

in Eqn. (1) consists of independent and identically distributed Gaussian random variables which represent the inherent noise found in gene expression.

When we define the functions  $s(\alpha)$  and  $d(\alpha)$  as given in Eqn. (2), we obtain  $M$  equilibrium solutions  $\bar{\mathbf{m}}^{(k)}$  of Eqn. (1) in the form of

$$\bar{\mathbf{m}}^{(k)} = [\bar{m}_1^{(k)}, \dots, \bar{m}_M^{(k)}]^T \quad k = 1, \dots, M$$

with components  $\bar{m}_i^{(k)}$ , see Eqn. (5).

$$\bar{m}_i^{(k)} = \begin{cases} \varphi(\alpha) & i = k \quad (H \text{ value}) \\ \frac{1}{2} [\sqrt{4 + \varphi(\alpha)^2} - \varphi(\alpha)] & i \neq k \quad (L \text{ values}) \end{cases} \quad (5)$$

The interpretation of the constant term

$$\varphi^* = \frac{1}{\sqrt{2}}$$

can be given as follows. For  $\varphi(\alpha) = \varphi^*$  the high and low values are equal, and therefore there is no preference for any solution in particular. This occurs, however, when  $\alpha = 0$ , so all  $m_i$  will be nearly equal and the adaptation toward a solution is done by random walk.

In summary, the general behavior of ARAS can be described as follows. The system in Eqn. (1) converges to solutions which have a single high value ( $H$ ) and all other values are low ( $L$ ). The dynamics of activity  $\alpha$  influences the selected values. When  $\alpha$  is high, the high value  $H$  also approaches 1.0, i.e., the selection becomes more deterministic. On the other hand, for small  $\alpha$ ,  $H$  and  $L$  become equal and the probabilities for selecting the next hop is controlled by the noise term, see Fig. 3.

### B. Activity Dynamics

In the original biological model, activity performs a mapping of the availability of the nutrients to a single real value. Thus, it controls the influence of the noise terms  $\eta_i$  on the dynamic behavior of the system. If  $\alpha \approx 0$ , the equations are dominated by  $\eta_i$  and the system essentially performs a random walk. On the other hand, for  $\alpha \approx 1$ , the random influence recedes and the system converges to an attractor. In this paper, we use the *packet delivery ratio* of a flow measured at the destination node as activity, although other types of mappings are also feasible, e.g., number of hops, path length, available bandwidth, or combinations of several factors [4].

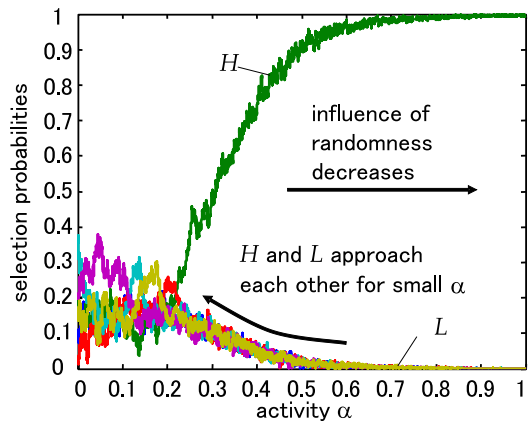


Fig. 3. Influence of  $\alpha$  on output probabilities

## IV. AD-HOC ROUTING WITH ATTRACTOR-SELECTION

So far we only discussed about the basic properties of attractor selection model to choose the next hop among the neighboring nodes and to react to updates of the network topology. Now, we will elaborate on how the selection method can be implemented within a stable and robust ad hoc routing protocol.

### A. AODV-like Reference Model

Basically, the main focus of our mechanism is to operate in a fully self-adaptive way while reducing the amount of overhead from flooding the network with probe packets to find new routes. We use an AODV-like method [17] as reference model in order to compare the efficiency of our proposed mechanism. This reference method operates as follows. When a new connection from a source node to an unknown destination node is requested, it broadcasts *route request packets* (RREQ) to its neighbors. On receiving a RREQ, a node responds to it with a *route reply* (RREP) if it has a route to the destination in its routing table, otherwise it rebroadcasts the RREQ to its neighbors. A node only processes the first RREQ and discards all subsequent RREQ of the same flooding attempt. This whole process is repeated until the destination node is found which then replies with a RREP to the source node along the reverse path. Upon reception of a RREP, all intermediate node store the next hop for that destination in their routing table.

In case that one of the forwarding nodes becomes unavailable, the previous hop node responds with a route error (RERR) to the source node, which then initiates another flooding attempt like described above.

### B. Route Setup Phase of Proposal

In our proposed method we want to limit the number of broadcasts and keep a simple method for selecting the next hop node in case a route breaks. Initially, the source needs to find the destination node, which is done in the exact way as described above for the AODV mechanism. However, instead of keeping routing table entries, node  $n$  now maintains a vector

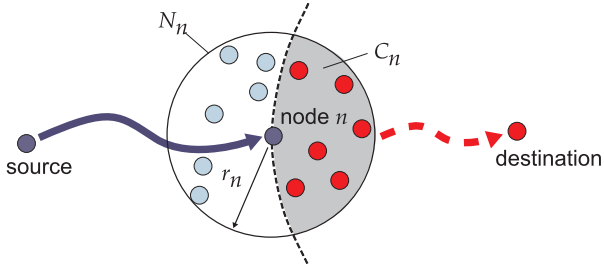


Fig. 4. Decision of next hop with ARAS

of hop probabilities to each of its neighbors

$$\mathbf{p}_n = [p_{n,1}, \dots, p_{n,M}] \quad (6)$$

when node  $n$  has  $M$  neighboring nodes. Probing neighboring nodes can be done by exchanging HELLO messages as in other routing protocols and the entries of neighbors which do not reply for a given period are removed and new neighbors are added when they appear within the transmission range. These probabilities are maintained for each connection between a source and destination pair and set up reactively when a route is requested. We choose the same route setup mechanism as in AODV. When the destination node sends a route reply (RREP) message back to the source on the reverse path and an intermediate node  $n$  receives a RREP, it sets up  $\mathbf{p}_n$  with  $p_{n,k} = 1$  if the next hop of  $n$  toward the destination is node  $k$  and  $p_{n,i} = 0$  for all  $i \neq k$ .

### C. Route Maintenance Phase

Once the route has been set up, data packets are transmitted over this route from the source to destination. Each intermediate node  $n$  chooses its next hop according to the probability vector  $\mathbf{p}_n$ . Whenever, the destination node receives a packet, it evaluates the current quality of this connection in terms of the activity term  $\alpha$  and piggybacks this information on the acknowledgment packets back to the source. All intermediate nodes update their hop probability vector according to the new activity. Basically, this means that the proposed method operates at first similarly to AODV and remains this way as long as the packet delivery ratio is near one. However, having a node disappear along the path causes that the delivery ratio drops and the lower it becomes, the more randomly the next hop selection is done. Nodes which receive data packets for a connection for the first time, set up the vector  $\mathbf{p}_n$ . This will cause that in the event of a route failure, many nodes will be setting up these vectors. In order to eliminate the unnecessary overhead of keeping unused routing vectors with stale connections in memory, the ARAS state levels  $m_i$  decay over time at a rate  $\delta$ .

$$\frac{dm_i}{dt} = \delta (0 - m_i) \quad i = 1, \dots, M \quad (7)$$

If all entries remain below a threshold for a certain period of time, the node has not been used for this connection and the vector is removed from memory. The vector  $\mathbf{p}_n$  evolves over time and is obtained from the system in (1) after normalization.

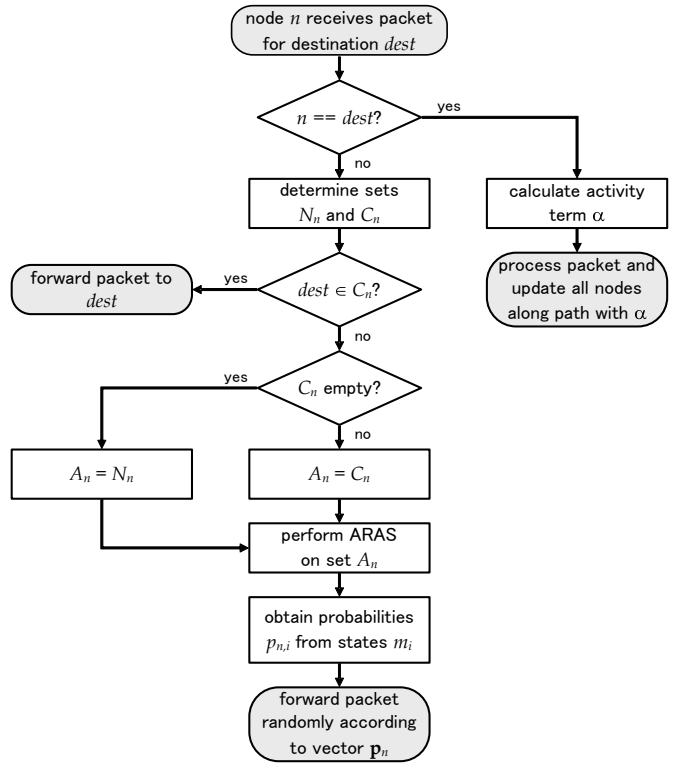


Fig. 5. Flowchart of basic ARAS-based routing mechanism

The reason for using a decay instead of a simple timeout is that this can be seamlessly integrated in the calculation of the ARAS state values.

The performance of this method can be further improved if the neighboring nodes are considered in two sets, a *neighbor set*  $N_n$  and a *candidate set*  $C_n$  of nodes which lie in the direction toward the destination, see Fig. 4. This, however, requires either some kind of position information obtained through GPS or the information of the relative position of a node to the destination, which can be obtained by additional signaling. In this work, we assume that the candidate set can be identified within the neighbor set. This can be done for example by considering the hop level at which a RREQ packet is received from the source node in the initial flooding stage. Nodes with a higher hop level than their neighbors are thus more likely to lie closer toward the destination.

We can now summarize the basic algorithm for packet forwarding with MARAS when node  $n$  receives a packet for destination node  $dest$ , see Fig. 5. After receiving the packet, the node checks if it is the intended destination. In this case, it determines the new activity  $\alpha$  (in our case the packet delivery ratio) which is then propagated to all nodes in the reverse direction along the path. If the node is not the destination, it queries its neighboring nodes and obtains from the replies the neighbor set  $N_n$  and candidate set  $C_n$ . With this knowledge the node  $n$  can see if the destination can be directly reached, i.e., the destination node lies in the candidate set. Otherwise, the candidate set, if not empty, is chosen as the set on which

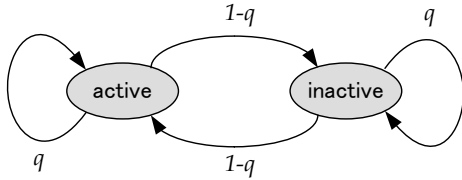


Fig. 6. State model of node activity

ARAS operates. However, due to the irregularity of the node distribution, it may occur that no suitable candidate exists in the direction to the destination. In order to avoid getting stuck in dead ends, the requirement of the next hop being nearer to the destination is relaxed and the entire neighbor set is used for ARAS. The ARAS equation in (1) is continually computed over time and based on the current state values  $m_i$  at a packet arrival, the selection algorithm is performed on the current set of available neighboring nodes. Finally, the resulting ARAS state vector is normalized to yield a probability distribution and the next hop is selected randomly following this probability distribution and the packet is forwarded to its next hop.

## V. NUMERICAL EVALUATION

In this section, we investigate the performance of the proposed method using ARAS for routing by simulation experiments. In order to focus on the behavior of the routing method, we assume idealized conditions for the MAC and PHY layer, i.e., no collisions, hidden nodes, etc. As a reference we use the AODV model as described in Section IV-A for comparison. Nodes are randomly distributed in a window  $W$  of unit size according to a homogeneous spatial Poisson process [18] with density  $\lambda$ . This means that on average there will be  $\lambda$  nodes in  $W$ . Obviously, the relationship between  $\lambda$  and the transmission range  $r$  of each node impacts the connectivity of the system. So, if there are too few nodes, the probability of being able to reach the destination and hence obtaining a sufficient packet delivery ratio will be very low, regardless of the considered routing mechanism. A brief theoretical discussion of this issue can be found in [4]. We use  $r = 0.2$  for all of our experiments and each simulation runs for a duration of 10000 time steps and is repeated 1000 times with a new random layout over which average values are taken.

In order to investigate the robustness and the reactive behavior of the routing method, each node has an active state of operation and an inactive (sleeping) state at which it is unable to forward any incoming packets. For the sake of simplicity, we consider discrete time steps, and the node activity is characterized by a probability  $q$  with which it remains in its current state at each time step, see Fig. 6. Thus, the active and inactive phases are characterized by geometrically distributed random variables. Note that since initially the same flooding mechanism is used to find the destination node in both methods, the results for  $q = 1$  (no activity change) would be nearly equal in both cases. In the following, we will consider the two values  $q = 0.9995$  and  $q = 0.995$ , which correspond

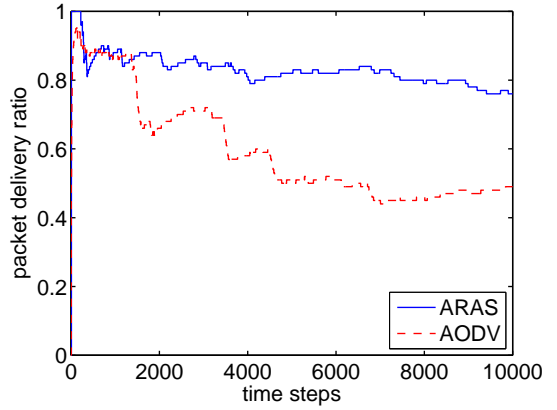


Fig. 7. Trace of packet delivery ratio from single run

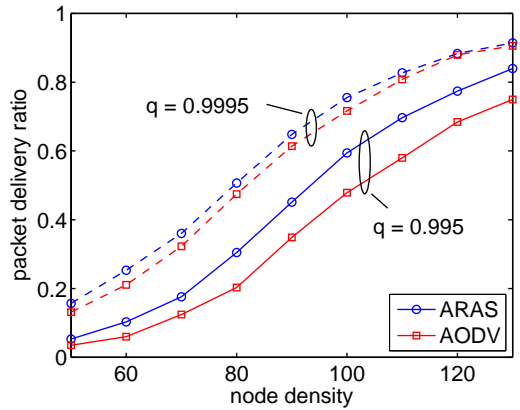


Fig. 8. Packet delivery ratio

to mean activity/inactivity durations of about 2000 and 200 time steps, respectively.

### A. Efficiency of Packet Delivery

At first we compare the efficiency of the routing method with ARAS to that of AODV in terms of the packet delivery ratio. We assume that no retransmissions of erroneous packets are made, so the effective packet delivery ratio may be higher when a retransmission mechanism on a higher layer (e.g. TCP) is applied.

In Fig. 7 the trace of the packet delivery ratio is depicted over time for an individual simulation run with identical conditions for both routing methods. We can see that when nodes become inactive along the path, the delivery ratio with AODV drops step-like over time. On the other hand, the proposed method shows a good, nearly constant behavior. The parameters used in this run were a node density of  $\lambda = 120$ ,  $q = 0.995$ , and radius  $r = 0.2$  with the same node layout and activity behavior of each node.

Fig. 8 shows the packet delivery ratio of both methods for  $q = 0.9995$ . We can see when  $q$  is high, both methods have nearly equal results. For smaller  $q$  which leads to shorter sojourn times in the activity/inactivity phases, the difference

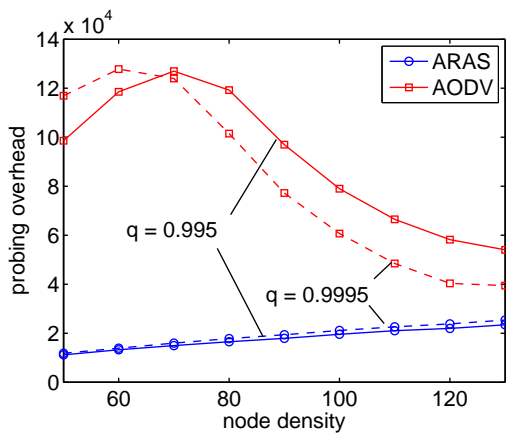


Fig. 9. Overhead from probe packets

between routing with ARAS and AODV becomes more evident. In both cases, the packet delivery ratio can be improved compared to AODV.

### B. Overhead from Probe Packets

When we look at the overhead due to probe packets until the path is found, we can see in Fig. 9 that the proposed method with ARAS shows a significantly better performance than the flooding based AODV. When the node density increases, AODV can find a path faster than when there are only few nodes. Note, however, that for AODV only those probe packets are considered until a path is found. Therefore, in an actual network, the total number of probe packets in AODV is much higher at large densities.

We can further see from Fig. 9 that the difference in node activity has hardly any influence on the number of probe packets needed for routing with the proposed method. In fact, the absolute number of packets can be further reduced since in this experiment the current neighborhood is evaluated at every time step. This is a kind of a worst case assumption. In practice, updating the neighbor set only at a certain time interval is sufficient enough. This would further reduce the number of probe packets in ARAS.

## VI. CONCLUSION

In this paper we presented a simple, yet efficient routing mechanism which is very robust to routing failures. It is based on the biological attractor selection mechanism and uses the concept of attractors to select among all candidate next hop nodes the one which is best in terms of a metric, in our case the packet delivery ratio. Our focus lies on robustness, but numerical results showed that the method also performs similarly well as the AODV reference model used in the simulations. However, when the node activity is subject to frequent changes due to their duty cycle, churn, or mobility, the number of required broadcasts can be reduced compared to AODV.

The experiments in this paper dealt only with static nodes, yet the method is also applicable when the nodes are mobile.

However, in this case the selection of the candidate set must be performed differently. In the future we wish to find more robust methods on limiting the set of candidate nodes among the neighbors which works well in mobile environments.

## ACKNOWLEDGMENTS

This research work was supported by “Special Coordination Funds for Promoting Science and Technology: *Yuragi Project*” and a Grant-in-Aid for Scientific Research (A)(2) 16200003 of the Ministry of Education, Culture, Sports, Science and Technology in Japan.

## REFERENCES

- [1] P. Mohapatra, J. Li, and C. Gui, “QoS in mobile ad hoc networks,” *IEEE Wireless Communications*, vol. 10, no. 3, pp. 44–52, June 2003.
- [2] 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*, vol. 1, no. 1, p. e49, 2006.
- [3] K. Leibnitz, N. Wakamiya, and M. Murata, “Biologically inspired self-adaptive multi-path routing in overlay networks,” *Commun. ACM*, vol. 49, no. 3, pp. 62–67, 2006.
- [4] —, “Self-adaptive ad-hoc/sensor network routing with attractor-selection,” in *Proc. of IEEE GLOBECOM*, San Francisco, CA, November 2006.
- [5] Q. Xue and A. Ganz, “Ad hoc QoS on-demand routing (aqor) in mobile ad hoc networks,” *J. Parallel Distrib. Comput.*, vol. 63, no. 2, pp. 154–165, 2003.
- [6] D. Johnson, Y. Hu, and D. Maltz, “The dynamic source routing protocol (DSR) for mobile ad hoc networks for IPv4,” IETF Network Working Group, RFC 4728, February 2007.
- [7] C. Perkins, E. Belding-Royer, and S. Das, “Ad hoc on-demand distance vector (AODV) routing,” IETF Network Working Group, RFC 3561, July 2003.
- [8] V. Park and S. Corson, “Temporally-ordered routing algorithm (TORA) version 1, functional specification,” IETF MANET Working Group, Internet Draft, July 2001.
- [9] S. Mueller, R. P. Tsang, and D. Ghosal, “Multipath routing in mobile ad hoc networks: Issues and challenges,” *Lecture Notes in Computer Science*, vol. 2965, pp. 209–234, Jan. 2004.
- [10] J. Broch, D. A. Maltz, D. B. Johnson, Y.-C. Hu, and J. Jetcheva, “A performance comparison of multi-hop wireless ad hoc network routing protocols,” in *Proc. of ACM/IEEE MobiCom ’98*, New York, NY, 1998, pp. 85–97.
- [11] S. Chakrabarti and A. Mishra, “QoS issues in ad hoc wireless networks,” *IEEE Communications Magazine*, vol. 39, no. 2, pp. 142–148, 2001.
- [12] C. R. Lin and J.-S. Liu, “QoS routing in ad hoc wireless networks,” *IEEE Journal on Selected Areas in Communications*, vol. 17, no. 8, pp. 1426–1438, August 1999.
- [13] C. Zhu and M. S. Corson, “QoS routing for mobile ad hoc networks,” in *Proc. of IEEE INFOCOM*, New York, June 2002, pp. 958–967.
- [14] L. Chen and W. B. Heinzelman, “QoS-aware routing based on bandwidth estimation for mobile ad hoc networks,” *IEEE Journal on Selected Areas in Communications*, vol. 23, no. 3, pp. 561–572, March 2005.
- [15] S. R. Medidi and K.-H. Vik, “QoS-aware source initiated ad-hoc routing,” in *Proc. of IEEE SECON*, Santa Clara, CA, October 2004, pp. 108–117.
- [16] K. Kaneko, *Life: An Introduction to Complex Systems Biology*. Berlin: Springer, 2006.
- [17] C. Perkins and E. Royer, “Ad hoc on-demand distance vector routing,” in *2nd IEEE Workshop on Mobile Computing System and Applications*, New Orleans, LA, Feb. 1999.
- [18] J. Kingman, *Poisson Processes*, ser. Oxford Studies in Probability. New York: Oxford University Press, 1993, vol. 3.