Abstract— Today’s network control systems have very limited ability to adapt to changing network conditions. The addition of reinforcement learning-based network management agents can improve Quality of Service (QoS) by reconfiguring the network layer protocol parameters in response to observed network performance conditions. This paper presents a closed-loop approach to tuning the layer three protocol based upon current and previous network state observations, specifically the HELLO Interval and Active Route Timeout parameters of the AODV routing protocol (AODV-Q). Simulation results demonstrate that the self-configuration method proposed here demonstrably improves the performance of the original Ad-Hoc On-Demand Distance Vector (AODV) protocol, reducing protocol overhead by 43% and end-to-end delay 29% while increasing the packet delivery ratio by up to 11% depending upon the traffic model.

Keywords— ad hoc, cognitive networks, reinforcement learning, wireless networks

I. INTRODUCTION

While the strict layering architecture of the Open Systems Interconnection (OSI) stack is conceptually useful, it is not as effective for wireless networks when time-varying traffic is served over a channel with limited throughput. Efficiently utilizing the resources with QoS provisioning requires a cross-layer optimization approach. As a result, better performance can be expected from information exchange across the protocol layers [27, 28]. The purpose of this paper is to address these issues by exploring the concept of intelligent network management for globally optimum performance in a dynamic wireless network deployment.

In typical network deployment scenarios, networks elements are limited in their abilities to adapt to changing application demands and topology characteristics, taking the context of these changes into account. In the case of routing in multi-hop wireless networks, battery-powered devices create challenging problems in terms of prolonging the autonomous lifetime of the network. In designing intelligent routing protocols, the various features of sensor networks lead to a set of optimization problems in routing path length, load balancing, consistent link management, and aggregation [1]. In real scenarios however, these factors are usually in conflict with one another, and influence the routing performance in a complex way. This in turn, leads to the need for a more sophisticated routing scheme that makes ideal trade-offs between multiple factors. Clearly, solving the optimization goals separately does not lead to a globally optimal solution; rather, all metrics should be addressed with respect to one another.

A solution for addressing these multi-variant optimization problems in network management lies in the vision of cognitive networks [2]. Cognitive networks continuously adapt to changing environmental conditions and/or user needs by constantly optimizing the bandwidth access and communication links. Typically, machine learning techniques, such as Q-learning [3] help implement the adaptation methods of self-configuration and self-management in the autonomic computing paradigm. Recent work further reinforces the efficacy of leveraging machine learning for network management task optimization [21,24,29,30].

The self-configuration of network systems has cross-layer ramifications for the protocol stack, from the physical (PHY), Medium Access Control (MAC), network, and transport layers to the application layer. Therefore, cross-layer design [4-7] approaches are critical for the efficient utilization of limited resources, to enable QoS guarantees, in future wireless and heterogeneous networks. In this paper, we present the concept of self-configuration in a cross-layer context, which can overcome the current limitations of network management in heterogeneous wireless networks, by allowing networks to observe, analyze and act [31] in order to optimize performance. Our approach is to augment the routing strategy of the AODV routing protocol with Q-learning, to ensure that the packet delivery ratio can be increased, while at the same time minimizing management overhead.

Toward the above stated goals, we present a new architecture of reconfigurable ad hoc routing management with Q-learning, namely, Q-Learning based Self-configuration (QLS) management and the AODV-Q protocol. The QLS management architecture enables nodes to efficiently learn optimal routing strategies, thereby enhancing the packet delivery ratio, end-to-end delay, and other QoS performance metrics. We present NS-2 simulation results showing that our cross-layer, self-configuration approach successfully improves the scalability of the AODV routing protocol in a heterogeneous network environment. The
 remainder of the paper is organized as follows: Section II presents a belief survey of related work. Section III gives an overview of our network architecture with reinforcement learning techniques for autonomic self-management. Section IV describes in detail AODV-Q. Section V explains the NS-2 simulation scenario. Section VI presents NS-2 simulation results. Finally, Section VII concludes by projecting future research directions.

II. RELATED WORK

A. Cross-Layer Approaches for Intelligent Network Management in Wireless Networks

The realm of network management covers a vast collection of issues, such as IP configuration, security and network monitoring. While these components are not unique to Mobile Ad hoc Networks (MANETs), they do become more difficult to optimize when nodal mobility, dynamic network membership, and unstable links are introduced into the network [8]. Depending on the speed of the Mobile Nodes (MNs), mobility can be classified into three categories: static, low mobility, and high mobility. The management layer of such a network should be able to take into account any of these three cases or combination thereof. In the case of low mobility, the steady-state performance should be optimized since incidental updates (e.g., for route discovery) can consume lots of resources. For high mobility networks, resource consumption, and delay due to route maintenance are important limiting factors [9].

Centralized network management architectures fail to provide effective scalability in MANETs. In the last few years, a distributed decision making scheme [10] has been introduced to address these concerns. In this proposed scheme, nodes may only be aware of their own neighbors and have no understanding of the size and extent of the network. Finding a mechanism that can deal with particular challenges associated with distributed decision-making in ad hoc networks is certainly non-trivial.

Recent research and present existing mechanisms do not provide a particularly good fit for a certain environment and an alternative paradigm is needed for a particular scenario such as field-based anycast routing using temperature field [11], a bidirectional abstraction to routing protocols for the asymmetric mobile network [12], network science based approaches for military applications [13] and context-aware protocol engine [14]. To cope with these demands management solutions based on cross-layer design [4-7] are necessary for efficient utilization of the limited resources in future wireless networks.

B. Challenges in MANETs

Several papers have classified MANET routing protocols in terms of their behavioral characteristics and applicability. We largely adhere to the standard convention of classification, namely flat, hybrid, and geographically-oriented protocols. Routing protocols which are not organized in any hierarchical fashion are commonly referred to as flat routing protocols [33]. Flat routing schemes have three main classifications: proactive (table-driven, e.g. Optimized Link State Routing Protocol (OLSR) [15]), reactive (demand-driven, e.g. Dynamic Source Routing protocol (DSR) [16], Ad-Hoc On-Demand Distance Vector protocol (AODV) [17]), and hybrid (e.g. Zone Routing Protocol (ZRP) [18]).

Dynamic Source Routing protocol (DSR) is a reactive protocol which uses source routing as a central mechanism [13]. When a route request (RREQ) is made by a particular node, it uses the destination route stored in its local route cache to send the data packet. Nodes along the path aggressively cache the path from the source node’s cache (which is embedded in the packet itself). However, if the node does not have the required route information cached, the route discovery process is initiated by flooding the network with route request packets. The request packets propagate throughout the network until they reach the destination node, or a node which has a cached path to the destination. The end node then sends a route reply with the newly discovered route source information back to the source node which then caches the path for future source routing. Further, destination nodes respond to all route request packets, thereby increasing the amount of aggressive caching taking place throughout the network.

The Ad-Hoc On-Demand Distance Vector (AODV) routing protocol is another routing protocol for multihop wireless networks, similar in nature to DSR. AODV shares DSR’s on-demand characteristics in that it also discovers routes on an as-needed basis via a similar route discovery process. However, AODV adopts a very different mechanism to maintain routing information. There is only one table entry per destination in any particular node’s routing table. AODV uses sequence numbers to determine the “freshness” of routes in the various routing tables. Without source routing, AODV relies on routing table entries to propagate the route reply (RREP) back to the source and, subsequently, to route data packets to the destination.

An important feature of AODV is the maintenance of timer-based states in each node with parameters (e.g. active route timeout, hello interval, etc) regarding utilization of individual routing table entries. A routing table entry is expired when not used recently. A set of predecessor nodes is maintained for each routing table entry, indicating the set of neighboring nodes which use that entry to route data packets. These nodes are notified by route error (RERR) packets when the next-hop link breaks. Each predecessor node, in turn, forwards the RERR to its own set of predecessors, thus effectively erasing all routes containing the broken link.
However effective AODV may be [39], it suffers from the following drawbacks in a mobile network environment:

(a) It does not frequently update the route to the destination.
(b) Due to the large Hello Timer values, there appears to be a periodicity in the route request generation which, in turn, can be attributed to poor link failure detection.
(c) It determines the ‘best effort’ shortest path, i.e. the shortest successful path.

In the case of proactive protocols, such as OLSR, there are sufficient exchanges of routing information to result in near-optimal routes. Therefore, OLSR is more resistant to packet drops at the MAC layer. However, one of the drawbacks of OLSR is that it generates routing traffic independent of application traffic [32]. Due to the higher routing overhead in proactive routing protocols, we chose the reactive routing protocol, AODV, in our cross-layer approach and focus on enhancing the protocol performance with a self-configuration mechanism.

To identify the trade-off issues when using reinforcement learning, it is crucial to study the impact factors of routing protocols, traffic load and mobility, and their impact on service delivery. A statistical design of experiments could be beneficial to identify both main effects and interactions of factors that best explain the response variables [19]. However in this paper the focus is on reconfiguring the critical timers, namely, hello interval and active route timeout (ART), to enhance network performance by dynamic context exchanges in heterogeneous networks.

C. Existing Protocol Parameter Tuning Solutions

Parametric tuning of routing protocols, and AODV in particular, has been of increasing interest in recent years [20, 34-38, 40]. In [20], Vadde and Syrotiuk explore the sensitivity of AODV protocol parameter tuning in conjunction with network performance metrics. They show that nodal mobility is the major contributing factor to delay, due to frequent route re-establishing processes. Additionally, they explore the fact that the packet arrival rate is the main contributing factor to affecting throughput, and that the timer interactions of network events and timers, such as the ACTIVE_ROUTE_TIMEOUT, directly affect the generation of performance-degrading protocol overhead packets.
Other works have proposed solutions on how to concretely modify these protocol parameters to improve network performance. In [36], Xing et. al. propose a modification to AODV, DA-AODV (Dynamically Adjusting AODV), which measures network scale to limit the scope of network max hop count. Leveraging the RREQ and RREP packets to carry this information, the max hop count indicates the number of max hops between source and destination pairs. Network max hop count is calculated on a per-node basis, indicating the max hops on a path for a particular source/destination pair. The authors add a new routing table parameter, Net_Diameter, to denote the max hop count value for each node’s routing table entry. When the routing table entry changes, Net_Diameter is compared against every table entry to ensure that it is set equal to the max hop value. When a node wishes to send a RREQ message to a particular destination, the source node first compares its max hop count with the Net_Diameter value, setting either of the two values to the greater of the two values, thereby allowing the HELLO packet to be broadcasted over the whole known network. By increasing the range of network discovery, the authors show a reasonable reduction in end-to-end delay and route error packets, due to the enhanced routed discovery mechanisms.

In [37, 38] we proposed a general framework for autonomic network management in heterogeneous network environments. Specifically, in [37] we proposed using Q-learning to load-balance OSPF traffic to avoid link congestion. This was achieved by having network agents observe and track queue length of nodes in various routes. The reinforcement learning agent would then compute and track these queue lengths over time to determine the optimal routes to facilitate network-wide load balancing, and resulting in dramatic decreases in packet loss. In [38] we provided initial experimental results of leveraging reinforcement learning to improve AODV routing protocol performance over standard AODV by tuning the HELLO Interval. This work was a cursory study of how sensitive AODV would be to parameter modification in a heterogeneous environment. However, this work did not compare these modifications of AODV to any existing AODV parametric tuning solutions, nor was the learning rate adaptation explored. Moreover, the approach we use in this paper for measuring application performance is more tightly coupled to the actual realities of the network behavior. By using the max observed end-to-end delay instead of a predefined max allowable end-to-end delay, as part of the feedback mechanism to determine which protocol parameters to tune, we now observe better results.

For the purposes of this work, we chose to compare the performance of AODV-Q to the modified protocols Modified-AODV (Mod-AODV)[34] and Optimized-AODV (Opt-AODV)[35], as well as standard AODV. Since both solutions dynamically update the HELLO interval, the modifications proposed in [34] and [35] are more closely related than other
existing proposed protocol enhancements. In the next section, we cover the details regarding our proposed protocol tuning enhancement, aided by autonomic/cognitive management principles, leveraging Machine Learning for better optimization and performance. After presenting comparative performance results, we then give the results of an initial investigation into

III. SELF-CONFIGURATION FOR AODV

A. Self-configuration parameters for AODV

In AODV, the hello interval and ART values are important parameters to cope with link failures caused by network dynamics. However, these timers are typically set in a trial-and-error manner or set at a constant value, which can lead to great inefficiencies with respect to performance [20]. We apply the Q-Learning technique, leveraging cross-layer performance, to identify any possible performance implications involving these timers.

AODV uses hello messages, periodic local broadcasts by a node to inform each mobile node in its neighborhood [21]. The hello messages may list other nodes from which a mobile has heard, thereby yielding a broader knowledge of network connectivity. Setting the optimal hello interval is a crucial aspect of maintaining network connectivity.

The route discovery process of AODV allows the intermediate nodes to store a route’s state between the endpoints [22]. Each node keeps this state for a length of time given by the ART parameter. Every time the route is used, the timer is reset back to the ART value. The ART is a static parameter that defines how long a route is kept in the routing table after the last transmission of a packet on this route. This parameter is arbitrarily set to 3 seconds. Comparatively speaking, DSR keeps a similar time-out parameter, denoted route cache timeout, but with a value set at 300 seconds.

The use of static values does not take into account either the actual lifetime of the path or the scale of the time correlation between two successive connections between the same end-points. Finding an optimal value requires a balance between choosing a short ART that causes a new route discovery, even if a valid route is still available, and choosing a long ART, which risks sending packets on an invalid route. In the first case, the cost is the initiation of a new route discovery that could be avoided, and in the second case it is the loss of one or more packets and the initiation of a RERR process instead of a new route discovery phase.

B. Q-learning in MANET Routing

The varied features of wireless networks lead to many optimization problems with respect to achieving specific performance objectives. The idea of applying reinforcement learning to routing in networks was first introduced by Boyan and Littman [23]. They showed that the Q-learning [3] based routing can compete with the shortest path algorithms, without prior knowledge of the network topology. Q-learning has also been applied to routing in ad-hoc networks [1]. Collaborative reinforcement learning (CRL) also introduced and evaluated in [24] as a self-organizing technique for building a MANET routing protocol. To the best of our knowledge, no existing routing scheme with reinforcement learning takes into consideration optimization goals (routing path length, load balancing, consistent link management, and aggregation) combined with a cross-layer approach.

C. Cross-Layer, Autonomic Network Management Architecture

The cross-layer architecture for our proposed cognitive management framework is explained in Figure 1. One of the main advantages of cross-layer design is to make protocols aware of current network state in a localized but distributed fashion. By introducing network and application layer context to the network management agents, this improves the higher-level processes of the middleware, allowing our QLS mechanism to exploit broader knowledge of the network state, and improve overall system performance.

Other proposals for implementation of cross-layer information exchange have been put forth in the current literature. These proposals can be categorized in three main groups [7]: (a) direct communication between layers, (b) a shared database across the layers, and (c) completely new abstractions. Specifically, we present the cross-layer model which sets performance expectations relative to performance observed thus far, in support of application-layer performance optimization, calculates reward and penalty values in the middleware layer, and uses those values to inform protocol parameter tuning decisions at the network layer. Figure 2 is an illustration of the cross-layer design approach which conveys how QLS in the middleware layer can interact with the other reconfigurable modules in the network layer. The following steps describe the detailed workflow of this management scheme.
Step 1: The management module at the middleware layer gathers application demands and determines the corresponding requirements (in this case, minimization of ETE delay).

Step 2: The Q-learning agent in the middleware layer receives the performance requirement in the form of reward and penalty formulas.

Step 3: At the network layer, the AODV protocol provides the Q-learning agent with the decision variables, including end-to-end delay, RERR and RREP.

Step 4: The Q-learning agent decides which action should be used to enhance performance.

Step 5: The Q-learning agent reconfigures the routing parameter(s) accordingly (hello interval and ART).

Step 6: Loop back to step one to iteratively observe the effects of environmental actuation and reformulate decision parameter values for Q-Learning agent based upon new observations.

Table I summarizes the proposed autonomic management approach, with respect to qualitative analyses of the challenges faced in MANETs.

IV. Q-LEARNING BASED SELF-CONFIGURATION (QLS) FOR AODV

A. Q-learning

In Q-learning [3], each time an action \( a \) is executed, an agent receives an immediate reward \( r \) from the environment. The agent then uses this reward and the expected long-term reward to update the Q-values, which in turn influences future action selection. Its simplest form, one-step Q-learning, is defined as:

\[
Q(s, a) = (1 - \alpha) \cdot Q(s, a) + \alpha \cdot \max_{a'} Q(s', a')
\]

where \( \alpha \) is the learning rate \( (0 < \alpha \leq 1) \), which models the rate of the updating of Q-values. The variable \( s \) represents the present state observation and \( s' \) the new state which the algorithm will explore. The variable \( a \) represents the action which led to state \( s \) and \( a' \) the action that leads to \( s' \). The Q-value itself is a numerical value which represents the current state action pair. In this context, the state is the current performance of the network and the action is how to tune various protocol parameters. Finally, \( Q(s, a) \) is the Q-value derived from the current state-action pair, and \( \max_{a'} Q(s', a') \) is the max q-value (reward) that can be obtained from next state \( s' \) over all possible actions \( a' \). As a model-free Reinforcement Learning technique, Q-learning requires no knowledge about the underlying reward or transition mechanism; thus it is applicable to the problem of learning routing strategy in ad hoc networks, where explicit state-space mapping can become computationally cumbersome. Specifically, mapping out the possible permutations of networks settings, nodal mobility, and traffic interactions would be potentially infeasible, and Q-Learning allows us to avoid this task by exploring the state space of local state-action pairs without globally mapping it.

B. Q-learning based self-configuration

In our implementation of AODV-Q, each node has two Q values, \( Q_{\text{penalty}} \) and \( Q_{\text{reward}} \). \( Q_{\text{penalty}} \) denotes the penalty Q value for unstable network status, which makes the node take the action of decreasing ART and Hello interval. \( Q_{\text{reward}} \) represents the stability reward of the network, which will make the node take the action of increasing ART and Hello interval. With respect to the Q learning calculation,

\[
Q_{\text{penalty}} = (1 - \alpha) \cdot Q(s, a)_{\text{penalty}} + \alpha \cdot Q(s', a')_{\text{penalty}}
\]

(1)
**TABLE I. CROSS-LAYER RECONFIGURATION IN WIRELESS MESH NETWORK MANAGEMENT**

<table>
<thead>
<tr>
<th>Key Challenges</th>
<th>Previous Approaches</th>
<th>Our Approach</th>
</tr>
</thead>
<tbody>
<tr>
<td>Route Management</td>
<td>- Routing protocols directly manage the path</td>
<td>- Reconfigure (Tuning) the routing protocols to discover more-optimal paths</td>
</tr>
<tr>
<td>Management Overhead</td>
<td>- Relatively fixed management overhead (e.g. proactive protocols (OLSR) have almost fixed overhead)</td>
<td>- Automatically (Adaptively) change the control message frequency according to the change of application demands</td>
</tr>
<tr>
<td>QoS management</td>
<td>- Resource management is limited to network layer - Mostly static, flat QoS support</td>
<td>- Cross-layer resource management from network layer to application layer - Dynamic QoS support by reconfiguring with Q-learning agent</td>
</tr>
<tr>
<td>Overall Performance</td>
<td>- Not aware of application demands</td>
<td>- Support Performance by our cross-layer approach. When reconfiguring routing parameters, end-to-end delay is used to determine the reward/penalty values as performance is more heavily dependent on end-to-end delay.</td>
</tr>
</tbody>
</table>

\[ Q_{reward} = (1 - \alpha) Q(s, a)_{reward} + \alpha Q(s', a')_{reward} \]  
\[ Q_{reward} = n \left( \frac{ETE_t}{ETE_{max}} \right) = n \left( \frac{ETE_{max}}{ETE_t} \right), \]  
\[ Q_{penalty} = n \left( \frac{ETE_t}{ETE_{max}} \right) \]

In AODV-Q, each node makes its self-configuration decision based on the local routing information, represented as the two Q values which estimate the quality of the alternative actions. These values are updated each time the node receives a RREP packet. The reward value is given by:

\[ \text{Reward} = Q(s', a')_{reward} = n \left( \frac{ETE_t}{ETE_{max}} \right) = n \left( \frac{ETE_{max}}{ETE_t} \right) \]

when a ROUTE REPLY packet reaches the source and there is a path from the source to destination. Further, the penalty value is calculated as:

\[ \text{Penalty} = Q(s', a')_{penalty} = n \left( \frac{ETE_t}{ETE_{max}} \right) \]

when a ROUTE ERROR packet is generated due to a broken route, or a ROUTE REPLY packet reaches the source and there is no path from the source to destination. In the above equations, ETE is the current end-to-end delay; ETE_{max} is the maximum end-to-end delay observed thus far. The value n denotes the normalization constant. The end-to-end delay is used to affect the reward and penalty because network protocol performance has been observed to be more tightly coupled with the ETE delay metric than other network performance metrics [20]. The algorithm for calculating the penalty/reward values and updating the ART and Hello interval is as follows:

1. For a period of time (“learning_phase_duration” = 30 seconds), increase or decrease the ART and Hello intervals by 2 with equal probability.
2. When receiving a successful ROUTE REPLY message, calculate the reward value, update \( Q_{reward} \), and GOTO step 4.
3. When receiving an unsuccessful ROUTE REPLY message or ROUTE ERROR message, calculate the penalty value, update \( Q_{penalty} \), and GOTO step 4.
4. If \( Q_{reward} > Q_{penalty} \), Decrease the ART and Hello Interval by 1, ELSE Increase each value by 1.

To avoid overly-aggressive changes in the ART or Hello Interval values, we converged upon a learning rate of 0.1. Future research will entail an investigation of the effects of dynamic learning rate tuning in response to observed performance thresholds. Section VII conveys cursory results of experimentation with respect to an adaptive learning rate scheme.

If MN A receives a RRER then node A is more likely to decrease ART based on its Q-learning agent. MN B is going to decrease the Hello interval because MN B receives RREP indicating no valid route exists. But MN C may increase its Hello interval when MN C receives RREP of successful route discovery. By virtue of distributed decision-making, the different nodes on a given path may have different timer values, allowing the intermediate nodes, which have different mobility patterns, to quickly and reactively reconfigure their routing parameters. This enhanced reactivity from our cognitive framework improves the stability of the generated routes as will be illustrated in the ensuing results discussion.
<table>
<thead>
<tr>
<th>Simulation Parameters</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>Simulation Area</td>
<td>• 2000m × 2000m</td>
</tr>
</tbody>
</table>
| Number of nodes       | • Mobile Nodes (MNs) = 100  
                           • 3 Wireless Gateway Nodes  
                           • 6 Fixed Nodes |
| Mobility Model        | • Random Waypoint  
                           • Speed (m/s) = Uniform (0, 15)  
                           • Pause Time (s) = 0, 60, 120, 180, 240 |
| Wireless Interface    | • IEEE 802.11b, 11Mbps |
| Wireless Transmission Range | • 350m |
| Traffic Flows: CBR    | • 5 MN sources  
                           • ↔ 2 Fixed Server and 3 Mobile Client destinations  
                           • Packet size: 15kB  
                           • Transmission rate = 10 pkts/s  
                           • Application Profile: Lo-Res Video traffic |
| Traffic Flows: TCP    | • 5 MNs source  
                           • ↔ 2 Fixed Servers and 3 Mobile Client destinations  
                           • Packet size: 512B  
                           • Application Profile: FTP traffic |
| Simulation Time       | • 10 min (for each run) |

V. SIMULATION ENVIRONMENT

The performance of our network system has been evaluated with the NS-2 simulation tool [25]. As shown in Table 2, the simulation network is defined in a flat terrain of 2000 × 2000 m with 100 mobile nodes, 3 MANET GWs, 2 BGP routers, and 6 Fixed Servers. Table II displays the summary of NS-2 simulation parameters. At the physical and data link layers, the 802.11b standard was used for analysis. The main purpose of the simulation scenarios was to provide a framework to compare the performance of AODV-Q, the solutions proposed in [34, 35], and standard AODV protocols. Results were averaged over 20 runs for each protocol, for each max pause time.

The traffic models used to gather the simulation results consists of Constant Bit-Rate (CBR) Video Conferencing and File Transfer Protocol (FTP) application traffic profiles. In this analysis, node mobility is assumed to be random (i.e., independently selected by each node using a uniform distribution) movement rather than group movement. The mobile nodes are assigned a maximum speed of 15 m/s. In the simulation scenarios, each mobile node changes its location within the network based on the “random waypoint” model; that is, the node randomly selects a destination, moves toward that destination at a speed not exceeding the maximum speed (15 m/s) and then pauses; this interval is known as pause-time. In order to calculate the impact of high mobility on the protocol overhead, pause-time ranged from 0 to 240 seconds in duration. It should be noted that a pause-time of zero represents the worst case scenario, in terms of high topological instability, as the mobile nodes are constantly moving during the simulation.

For the AODV-Q simulations, each training episode, from the beginning of the simulation to 30s, each node randomly chooses actions decreasing or increasing active route timeout and hello interval. During the simulations AODV-Q reconfigures active route timeout between 3s and 10s, and hello interval between 1s and 10s. Table III conveys a summary of the results of the simulation, with the value of each performance parameter averaged across pause time runs, with percentage improvement over standard AODV listed to the right of the value in parentheses.
The performance of AODV-Q was evaluated in terms of responsiveness, protocol overhead, and packet delivery ratio. The performance results are compared with those derived under the standard AODV routing mechanism, as well as the solutions proposed in [34] (Mod-AODV) and [35] (Opt-AODV).

### A. Responsiveness

The network responsiveness resulting from decisions QLS applied to AODV was evaluated in terms Route Discovery Time and End-to-End Delay. The route discovery time, measured in seconds, is the measure of how long the protocol takes to determine a valid route once a request has been made. Figure 3b conveys the average measure of route discovery time for all four protocols. Opt-AODV tracks more closely with the standard AODV implementation with a 9% reduction in route discovery time, whereas AODV-Q and Mod-AODV offer consistent improvement in route discovery time (33% and 27% reduction respectively). The reduction in route discovery time can be attributed AODV-Q and Mod-AODV’s more temperate approach to tuning the Hello interval. Specifically, Mod-AODV uses thresholding to tune the hello interval by 25% at most (choosing a deviation fraction of 0.75, 1, or 1.25). Rather immediately adding or subtracting values to the Hello interval. And while AODV-Q does add or subtract values to the HELLO interval, it does so using a thresholded machine-learning based approach, whereby previous Q-values are taken into account to prevent radical swings in interval values. This approach leads to a more stable and accurate depiction of network dynamics than the approach used by Opt-AODV, which adds or subtracts values based upon immediate observation.

End-to-end delay, measured in seconds, is the measure of the time taken for a packet to be transmitted from the source and received at destination node. As delay is a good measure of the fitness of routes being selected, end-to-end delay was used as another measure of network responsiveness due to the decisions made by the routing protocols. Figure 3a displays the results of end-to-end delay measurements for all four protocols. While all four protocols generally reduced delay as network stability improved with increased pause time, using standard AODV as the performance baseline, AODV-Q and Mod-AODV exhibited 29% and 23% reductions in delay on average, respectively, whereas Opt-AODV showed a 10% reduction. Overall, end-to-end delay reduction can be attributed to the fact that AODV-Q, Mod-AODV, and Opt-AODV are tending to reduce protocol

### VI. PERFORMANCE EVALUATION

<table>
<thead>
<tr>
<th>Protocol</th>
<th>ETE Delay (s)</th>
<th>Route Discovery Time (s)</th>
<th>Routing Traffic (packets)</th>
<th>Route Errors (packets)</th>
<th>UDP Packet Delivery Ratio</th>
<th>TCP Packet Delivery Ratio</th>
</tr>
</thead>
<tbody>
<tr>
<td>AODV</td>
<td>0.104</td>
<td>0.846</td>
<td>89068</td>
<td>10013</td>
<td>0.764</td>
<td>0.89</td>
</tr>
<tr>
<td>AODV-Q</td>
<td>0.074 (-29%)</td>
<td>0.562 (-33%)</td>
<td>50047 (44%)</td>
<td>5378 (-46%)</td>
<td>0.846 (11%)</td>
<td>0.93 (5%)</td>
</tr>
<tr>
<td>Mod-AODV</td>
<td>0.08 (-23%)</td>
<td>0.61 (-28%)</td>
<td>59134 (-34%)</td>
<td>8018 (-20%)</td>
<td>0.794 (4%)</td>
<td>0.92 (3%)</td>
</tr>
<tr>
<td>Opt-AODV</td>
<td>0.094 (-10%)</td>
<td>0.768 (-9%)</td>
<td>63181 (-29%)</td>
<td>8413 (-16%)</td>
<td>0.781 (2%)</td>
<td>0.91 (1%)</td>
</tr>
</tbody>
</table>
overhead over time through tuning of the HELLO Interval (see figure 4a). Reduced traffic in the wireless medium allows the QLS scheme to realize a shorter queuing delay, resulting in shorter end-to-end delays.

B. Routing Overhead

Control overhead is measured in terms of the number of control messages generated by the routing algorithm. Figure 4a illustrates the number of routing control messages generated or relayed in the network. The standard AODV mechanism generates a greater number of control messages than does AODV-Q, Mod-AODV, and Opt-AODV, with reductions at 43%, 33%, and 29% respectively. This, in turn, translates into a higher probability of lost control messages in AODV due to collisions in the wireless medium. Consequently, routing paths are less reliable under standard AODV. All three protocols make reasonable progress toward overhead reduction, but the ability of AODV-Q to retain stable routes by tuning the active route timeout parameter yields further reduction in unnecessary route discovery traffic.

In simulations, the Q-learning agent at each node self-configures the active route timeout and hello interval according to the Q-value. Due to the distributed self-configuration of these parameters, the nodes send RREQs more appropriately to account for failed routes, improving the route freshness and the link failure detection processes. The route error is evaluated as the average number of RERR packets per second. As shown in figure 4b, Mod-AODV and Opt-AODV yield a RERR reduction of 20% and 16% respectively, whereas AODV-Q yields a RERR reduction of roughly 46%. While the previous two protocols have the ability to tune the HELLO message interval to alleviate unnecessary congestion and produce better routes, AODV-Q has the added advantage of being able to also tune the active route timeout interval. This allows the protocol to hold onto routes longer than the other three protocols when network stability is perceived over time. This behavior tends to favor stable routes being used longer, hence the reduction in route errors over time.

C. Packet Delivery Ratio

The third criterion we use for evaluation is that of the packet delivery ratio, which is the number or transmitted packets divided by the number of received packets. The delivery ratio was measured with respect to the two traffic types that traverse the wireless portion of the network, constant bit rate (CBR) UDP video traffic, and TCP-based FTP traffic. Figure 5a conveys the results for packet delivery ratios for UDP video traffic. While Mod-AODV and Opt-AODV yielded a 4% and 2% improvement respectively, AODV-Q showed an 11% improvement in delivery of video traffic. The tendency of AODV-Q to hold onto more stable routes in addition to the reduction in protocol overhead contributed to the larger percentage improvement for video traffic. Video conferencing applications generate packets with very short inter-arrival times, prompting the Q-learning agent at each node to self-configure a shorter active route timeout and hello interval. AODV shows large video conferencing packet delay variations due to its lack of efficiency and timeliness in finding new routes. AODV also shows larger end-to-end delay for video conferencing packets when the mobile nodes are highly dynamic, such as when the pause time is zero.

Figure 5b illustrates the packet delivery ratio for FTP traffic governed by TCP congestion control. In this case, the improvement upon the standard AODV implementation was less pronounced, with AODV-Q, Mod-AODV, and Opt-AODV
yielding 5%, 3% and 2% performance improvements respectively. TCP’s built-in congestion control can claim some responsibility for the higher packet delivery ratios of all protocols, TT the constant bit rate video traffic was sent regardless of congestion or loss feedback account for most of the difference in performance between the two traffic models. However, in general, we do see a larger performance improvement with AODV-Q due to the increased longevity of stable routes.

D. Impact of Learning Rates

In the preceding experiments, we executed the simulations with a relatively low learning rate ($\alpha = 0.1$). To analyze the impact of the learning rate itself upon the observed performance enhancement of QLS, we present the results of simulation scenarios with three learning rates: 0.01, 0.05, and 0.1.

First, we denote the various traffic types by index number, delineated by the mobility of the nodes involved:

- Profile 1: Ethernet-to-MANET
- Profile 2: MANET-to-Ethernet
- Profile 3: MANET-to-MANET

Fig. 6c conveys the fact that higher learning rates improved the packet delivery ratio for MANET Traffic Profile #3 (MANET-MANET). However, we found that lower learning rates were more advantageous for Traffic Profiles 1 and 2, with respect to improving their localized delivery ratios. We infer from these results that higher learning rates are more advantageous in highly mobile environments, where quick adaptability can enhance network performance. Further, such high learning rates could be harmful to semi-static components of the network, in that either the source or the destination is relatively stable with respect to the rest of the network.

Additionally, we devised a dynamic self-reconfiguration mechanism for determining the learning rate in real time. The learning rate updates occur each time a node agent computes its respective Q-values. If one of the following two events occurs, then the learning rate $\alpha$ is given by:

1) $\alpha = \alpha / 2$: Occurring when an agent receives a reward (the ROUTE REPLY packet reaches the source and there is a path from the source to destination).

2) $\alpha = \alpha * 2$: Occurring when an agent receives a penalty (a ROUTE ERROR packet is generated or a ROUTE REPLY packet reaches the source but there is no path from the source to destination).

As shown in Fig. 6c, self-reconfiguration of the learning rate enhanced the packet delivery ratio for Traffic #1 (MANET-MANET) and Traffic #3 (Ethernet-MANET). For traffic profile #2 (MANET-Ethernet) the self-reconfiguration of the learning rate was not as effective.
layer capability can meet the upper layer requirement.

...these consistent QoS support, the MAC layer could provide the essential context. The MAC layer could provide an indication of the network congestion level and achievable data rates; these calculations could, in turn, be used to determine whether the lower layer capability can meet the upper layer requirement.

One solution to the optimizing the learning rate is to use Bayesian exploration [26], to be explored in our future work) to tune and optimize learning rate values with respect to network performance. There further exists a need to investigate the optimization accuracy and the process of reward value assignment in the Q-value computation, in addition to the selection of correct parameters for self-configuration.

VII. CONCLUSIONS AND FUTURE RESEARCH

In this paper, we described a proposed framework for autonomously reconfigured network systems with a cross-layer approach. AODV-Q has been proposed to improve the performance of AODV, through the use of iterative network state observation. This is applicable to large heterogeneous networks, where the characteristics of the mobile nodes and application demands are different. We also presented experimental results. The performance results confirm that QLS dramatically reduces the protocol overhead compared to the standard AODV. AODV-Q achieves a higher packet delivery ratio while incurring shorter queuing delays. Specifically, with AODV-Q, it is possible to achieve shorter end-to-end delays while reducing the incidence of lost data packets. Therefore, the proposed autonomous self-configuration mechanism successfully improves the scalability and adaptability of the original AODV protocol in a heterogeneous network environment.

The work in this paper highlights some interesting and potentially important areas for future work, enumerated below.

A. Proactive vs. reactive network management

There is a fundamental tradeoff between proactive and reactive routing protocols, in terms of delay and control overhead. A proactive routing protocol generates routing traffic independent of application traffic. Due to the higher routing overhead in proactive routing protocols (e.g. OLSR), we have chosen the reactive routing protocol, AODV in our cross-layer approach and tried to enhance the protocol performance with QLS. However, it is inevitable that certain static networks will have especially high QoS demands which require the use of proactive routing. How to use proactive routing while minimizing the network-layer overhead is of key interest.

B. Performance evaluations in various network environments

It is important to verify the suitability of our approach to other heterogeneous networks (e.g. 3G, WiMAX, LTE and optical networks) with various traffic models and mobility models. It could be useful to provide results as a combination of larger networks and nodes.

C. Cross-layer design for heterogeneous application traffic with QoS guarantees

To achieve desired QoS guarantees it would be critical to consider the change of user demands in the application layer. For consistent QoS support, the MAC layer could provide the essential context. The MAC layer could provide an indication of the network congestion level and achievable data rates; these calculations could, in turn, be used to determine whether the lower layer capability can meet the upper layer requirement.
REFERENCES


