Presented at WWASN2006

Automatic Learning-based MANET Cross-Layer Parameter Configuration

Karen Zita Haigh
BBN Technologies
10 Moulton Street
Cambridge, MA 02138
khaigh@bbn.com

Srivatsan Varadarajan, Choon Yik Tang
Honeywell Labs
3660 Technology Drive
Minneapolis, MN 55418
{srivatsan.varadarajan, choon.yik.tang}@honeywell.com

Abstract

Mobile ad hoc networks (MANETs) operate in highly dynamic environments with limited resources. Current approaches to network configuration are static and ad-hoc, and therefore frequently perform extremely poorly. We describe our approach to network configuration control that relies on automatically learning *the relationships among configuration parameters and maintains near-optimal configurations adaptively, even during highly dynamic missions. We present a case study demonstrating the feasibility of the approach.*

Keywords: *Auto-Configuration, Cross-layer Parameter Adaptation, Learning, Mobile, Wireless*

Technical Area: *(1) Network Parameter Configuration, (2) Heterogeneous networks*

1 Introduction

Mobile ad hoc networks (MANETs) operate in highly dynamic, infrastructure-less and potentially hostile environments, with limited bandwidth and energy resources. Thus, it is desirable to adaptively allocate these resources, so that network-level performance requirements are met—in spite of inherently unreliable wireless channels and ever-changing network topology.

This paper describes an approach to intelligent tuning of protocol stack parameters to automatically configure each node in the MANET. ORACLE, the Optimizing Rapidly Adaptive Configuration Learning Engine, is a unique *hybrid* approach to network configuration control, combining Machine Learning and network modelling. Analytical network models capture useful general principles, but are incomplete, incorrect, and static. Traditional Machine Learning approaches reflect actual operating conditions, but poorly transfer knowledge to new domains and objective functions.

Our approach to MANET configuration relies on *automatically learning* the relationships among parameters. It maintains near-optimal configurations adaptively, even during highly dynamic missions. Our approach tunes parameters in a fully distributed manner so that a centralized processing node is not needed and communication overhead and delays are minimized.

In this paper, we describe the mathematical problem formulation for MANET configuration, present the general ORACLE approach, and finally present empirical results demonstrating the feasibility of the approach.

1.1 Comparison with Related Work

Adaptive techniques have been applied to improving network performance with some success. However, the points outlined below limit their utility.

One parameter: Most prior approaches adjusted a single parameter, e.g. data transmission rate [5, 12, 16], congestion window [4, 13, 17], and frame length [18]. An exception is Ye *et al* [27], which optimized large numbers of network parameters; however it is completely off-line and non-mobile.

One objective function: Most previous work designed models that capture parameter interactions for only one objective function, such as transmission errors [9], routing [8, 11, 24, 25] and power consumption [6, 23, 28]. Our approach does not depend on a single *a priori* objective function.

Model-based design: The most notable drawback of most approaches tried in MANET is that they are *hand built* models of the interactions among parameters. This approach to network configuration is not maintainable, particularly as protocols are redesigned, new parameters are exposed, or the objective function changes.

Mobility: Scalable approaches that rely on learned models of the parameter interactions were not implemented on mobile networks (e.g. adaptive routing [1], reconfigurable links [20], network parameter optimization [27]). Applications and protocols developed for the fixed, wired environment do not adapt transparently to the mobile, wireless environment [2].

ORACLE's hybrid learning methods for adaptive network configuration may be the only approach that tunes parameters across multiple layers in the protocol stack, with fully distributed local control and decision making, in a mobile ad-hoc network.

2 Problem Formulation

Consider a MANET having N heterogeneous nodes; each node i has a set of m_i *control parameters*, denoted $\mathbf{x}_i \triangleq (x_{i1}, x_{i2}, \ldots, x_{im_i})$, e.g. data rate at the PHY layer, maximum number of retransmissions at the MAC layer, and hello interval for neighbourhood discovery at the routing layer. Control parameters are intentionally exposed by the protocol stack for tuning; cross-layer issues are implicitly captured by selecting control parameters from multiple layers. Each node i also has a set of n_i *observables*, denoted $\mathbf{y}_i \triangleq (y_{i1}, y_{i2}, \ldots, y_{in_i})$. Observables include context that can be observed, such as throughput, latency, network topology, application start/end and mission context. Note that there may be *unobservable* contextual information, denoted $\mathbf{z}$. To capture changes over time, denote $\mathbf{x}_i(t)$ to be the value of $\mathbf{x}_i$ at time t, and $\mathbf{x}_i(t,...,t') \triangleq (\mathbf{x}_i(t), \mathbf{x}_i(t+1), \ldots, \mathbf{x}_i(t'))$; do likewise for $\mathbf{y}_i$ and $\mathbf{z}$.

Associated with the MANET is a real-valued scalar measure $J(t)$ that characterizes global, network-wide performance at time t. $J(t)$ could measure some combination of throughput, latency, mission requirements, user needs and other relevant factors. This measure is assumed to be a function f of all the control parameters, observable parameters, and unobservable factors: $J(t) = f(\forall i \in N(\mathbf{x}_i(0,\ldots,t), \mathbf{y}_i(0,\ldots,t-1)), \mathbf{z}(0,\ldots,t))$. An exact analytical expression for f is difficult to obtain, due to unobservable factors and complex cross-layer and cross-node interactions.

The ultimate goal is to solve the following distributed optimization problem: Design a fully distributed algorithm where every node i determines its control parameter values $\mathbf{x}_i(t)$ using only its own previous control values and observables $\mathbf{x}_i(0,\ldots,t-1), \mathbf{y}_i(0,\ldots,t-1)$ such that $J(t)$ is maximized for each t, despite the lack of an exact analytical expression for f. The algorithm design includes selecting observables; e.g., it may be useful to share the previous control settings of nearby nodes ($\mathbf{x}_{j \neq i}(t' < t)$).

3 ORACLE Approach

ORACLE builds a model of the *performance surface* to predict MANET effectiveness as a function of observables and control settings. Each node i builds a local, memory-less approximation of f, $\hat{J}_i(t) = \hat{f}_i(\mathbf{x}_{i(t)}, \mathbf{y}_{i(t)})$, simplifying the problem by assuming that decisions made by nearby nodes will be observable in $\mathbf{y}_i$. (For example, if a neighbour increases data rate, the node will observe increased congestion.) To configure the MANET, each node selects control values that optimize performance on this surface at time t: $\mathrm{argmax}_{\mathbf{x}_{i(t)}} \hat{f}_i(\mathbf{x}_{i(t)}, \mathbf{y}_{i(t)})$. (Note that the controller on each node i will be different.)

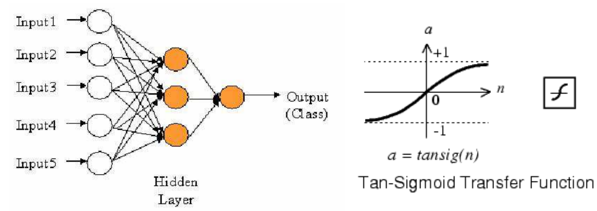

Figure 1. (a) Generic ANN; (b) a transfer function used in each node of the hidden layer.

We will first describe the techniques ORACLE uses to model the performance surface and then describe issues related to training the model.

3.1 Modelling the Performance Surface

We used artificial neural networks (ANNs) [15, 21] to learn the models $\hat{f}_i$. ANNs effectively handle discontinuities in the performance space, outliers in the data, and models with unknown functional forms. Once trained, ANNs calculate the function quickly, ensuring their utility in the MANET environment. An example is shown in Figure 1; each node in the hidden layers has a transfer function shown on the right.

Observables and control parameters form the inputs to the ANN. Observables may include mission traffic, channel gains, fading parameters, and noise levels. Control parameters may include transmit power, data rates, and modulation schemes. The output of the ANN is the predicted MANET effectiveness. MANET effectiveness metrics could measure combinations of factors such as throughput, battery life, latency, and application quality of service. The case study below describes the exact implementation of the ANN for our experiment.

A significant challenge for ORACLE is to accurately model the *extremely large search space*—perhaps as many as 1000 control parameters per node. To address this challenge, we designed a hybrid learning approach that leverages existing analytical models to learn only the *error* in the analytical models (shown in Figure 2)—thereby tackling a more feasible problem. This approach enables ORACLE to effectively capture the rich complexity of the domain and transfer learned knowledge from one environment to another. Given that analytical models are relatively rare and specific to a problem, ORACLE can use models in multiple places with different forms and advantages:

- Statespace reduction: e.g. relevant parameters, current operating space
- Feature construction: e.g. coarse estimate of MANET effectiveness, cross-layer interactions
- Within learned models: e.g. changing the form of the model
- Guidance for optimization: e.g. constraints; prin-

cipal variables

In the experiments below, constructed features provided a coarse estimate of global MANET throughput.

A key design requirement for MANET is to limit the amount of control knowledge that is shared among nodes. In the experiment, the only information shared among nodes is the measure of global MANET effectiveness. *All* other observables and control values are *local* observations on the node. Each node learns a model of how local observables and local control parameters affect global MANET effectiveness.

4 Case Study

We conducted an experiment to demonstrate the feasibility of ORACLE's learning approach to the MANET configuration problem. Our goal was to have each node independently observe *local* operating conditions and select the best parameter values to optimize *global* MANET performance.

4.1 Experiment Description

We performed our experiments in OPNET using a simplified Lakehurst scenario (a testsite now commonly used in American MANET research [3, 22]). Six vehicles (nodes) moved in a ring of five waypoints around a stationary command centre, as shown in Figure 3. We simulated a four-stage battle, with different mobility and communication parameters in each stage, as shown in Table 1. We used 802.11 MAC and the AODV routing protocol.

We built one learning controller for each node. The controllers used local information to decide control settings across multiple layers of the stack: Network Layer: Hello advertisement interval at 1, 4, and 8 seconds; MAC: maximum number of retransmissions at 2,4,8; PHY: transmit power levels implicitly controlled in 802.11b by varying data rates of 1, 2, 11 MBps.

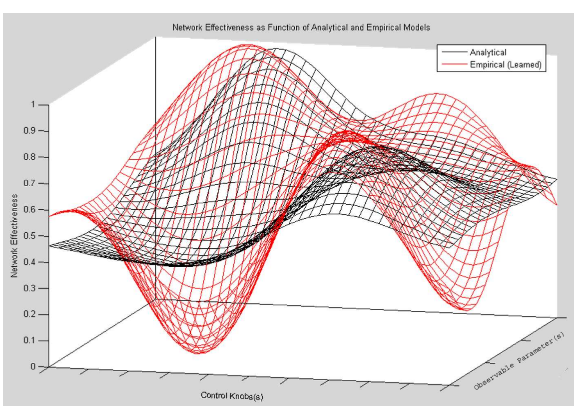

Figure 2. The performance surface as a function of n dimensions of observable parameters and m dimensions of control parameters. Analytical models guide the empirical learner, speeding the learning process.

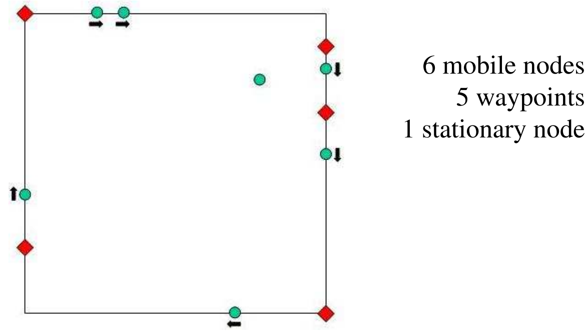

Figure 3. Simplified Lakehurst scenario.

Phase	Mobility / Data
1 - deploy	No motion, 1024 byte packets, constant bit rate (CBR)
2 - shape	Slow mobility (5 minutes between waypoints), 100 byte packets, CBR
3 - decisive ops	Fast (1 minute), 100 bytes, CBR
4 - consolidate	No motion, 1024 bytes, CBR

Table 1. Simplified Lakehurst experiment: mobility and communication parameters.

4.1.1 Training Data.

We collected 117 files of training data, of the $3^{3\times7} = 10.4$ billion possible configurations. The files consisted of the 27 homogeneous cases (i.e. nodes have identical parameters) and 90 heterogeneous cases. Each node collected these local statistics:

Application: velocity, heartbeatrate, packet size

AODV: total route requests sent, total route replies sent, total route errors sent, route discovery time

MANET: traffic sent (bits/sec), traffic received (bits/sec), delay (secs)

Radio receiver: bit errors per packet, utilization, throughput (bits/sec), packet loss ratio, busy, collision status

Radio transmitter statistic: busy

Wireless LAN: Control traffic received (bits/sec), control traffic sent (bits/sec), data traffic received (bits/sec), data traffic sent (bits/sec), delay (sec), dropped data packets (packets/sec), media access delay (sec), throughput (bits/sec), retransmission attempts (packets)

4.1.2 Experimental Procedure.

ORACLE's goal was to optimize message global performance, as measured by **MANET throughput**[1], calculated by the command centre. This throughput is the only non-local observable used by the learners.

We built an ANN *for each node*, as described in

[1]MANET throughput is the message traffic only, and does not include control traffic. Given that latencies could cause packets to 'accumulate,' we used a five-second cumulative total to mitigate measurement error.

Section 3. The inputs to the ANN were the 26 statistics listed in Section 4.1.1, plus location information as described below for each experiment. The output was MANET global throughput. Each node learned a model of how *local* observables and control parameters affect *global* performance. Training data consisted of the 117 files described in Section 4.1.1— 70% to train the ANNs, 10% to test them, and 20% to validate them.

Finally, the ANNs controlled a test run. The ANN on each node observed local conditions and selected the control values that predicted the highest global MANET throughput. (Note that the ANNs did not change during the run, and hence calculated values extremely rapidly.)

In Experiment #1, below, we demonstrate that a completely distributed learning approach improves performance over common alternate approaches. In Experiment #2, we explored issues of knowledge transfer, and demonstrate that a hybrid learning approach performs better than the basic learner.

4.2 Experiment #1: Learner compared to standard approaches

The first experiment asked whether the configuration problem could be solved through a learning approach, comparing a Batch Decision Learner (BDL) with the two most common approaches to configuring a MANET. The *optimal static homogeneous configuration* was the training configuration that generated the highest throughput during scenario Phase 3 (decisive ops); each node had the same configuration settings and the configuration did not change during the scenario. The *omniscient, omnipotent human "red team"* knew the mobility patterns and communication propagation properties of the environment, and could set heterogeneous configurations for the control parameters at each time stamp in the scenario.

The BDL ANNs used the statistics listed in Section 4.1.1 as input, plus each node used its *current node position (x,y)*. Figures 4 and 5 show the quality of the learned models for two nodes. The x-axis shows the actual throughput for the current observations, and the y-axis shows the estimated throughput. Mobility is clearly a factor in the ANN's ability to model the environment.

Table 2 and Figures 6 and 7 compare the results of a dynamic learning system to the static homogeneous configuration and to the best dynamic "red team" configuration. The test environment was identical to the training environment. The learning algorithm outperforms both the human red team and the static homogeneous configuration (except for the highly optimized Phase 3).

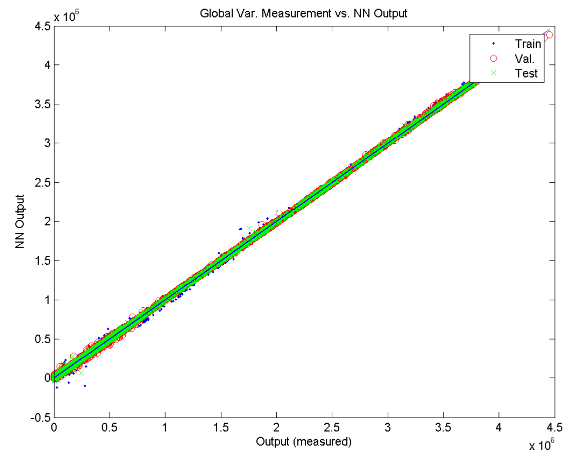

Figure 4. Model accuracy for a stationary node.

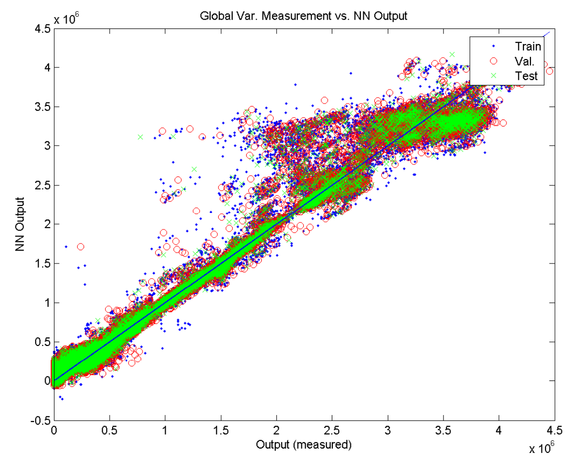

Figure 5. Model prediction accuracy for a mobile node.

4.3 Experiment #2: Knowledge Transfer

The second experiment tested knowledge transfer. We compared performance of the BDL with a Hybrid Decision Learner (HDL) when the training and testing environments are different. The hybrid learner combines analytical models with empirical models.

We built a BDL and an HDL ANN for each MANET node. Training data consisted of the statistics listed in Section 4.1.1, plus each node knew the *identity* and *distance* to its closest three neighbours[2].

HDL had an additional feature representing an analytical model of global MANET throughput using only locally observable information; this model attempts to capture routing issues in the MANET. Each node i estimates global throughput $\hat{T}_G$ according to: $\hat{T}_G(i) = 2 \times \sum_{j=1}^{j=7}(\hat{T}_{ij} \times P_{j7})$ where

- $\hat{T}_{ij}$ is an estimate of the throughput from node i to

[2]Identity is already known through standard routing protocols; distance is calculated by sharing current location.

Phase End	BDLearner	Dynamic RedTeam		Static Homogeneous	
1	1,470,136,320	1,376,018,432	94%	929,852,352	63%
2	520,424,320	375,285,152	72%	491,068,800	94%
3	96,661,600	72,932,000	75%	97,412,864	101%
4	1,350,628,704	1,086,611,456	80%	930,668,544	69%

Table 2. MANET throughput for the three control approaches. BDL performed notably better than the dynamic red team and static homogeneous configurations. Percentages reflect performance compared to BDL.

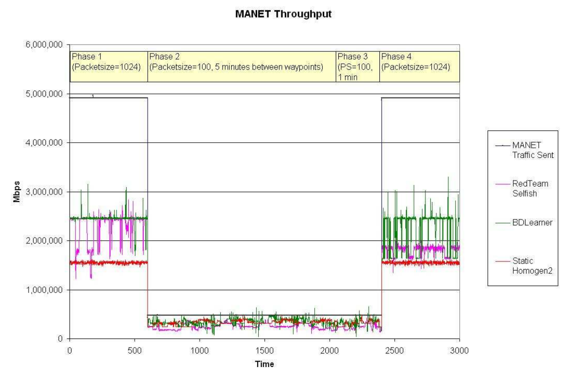

Figure 6. MANET throughput for three control approaches. This experiment shows that learning outperforms both the optimal static setting and a dynamic human expert.

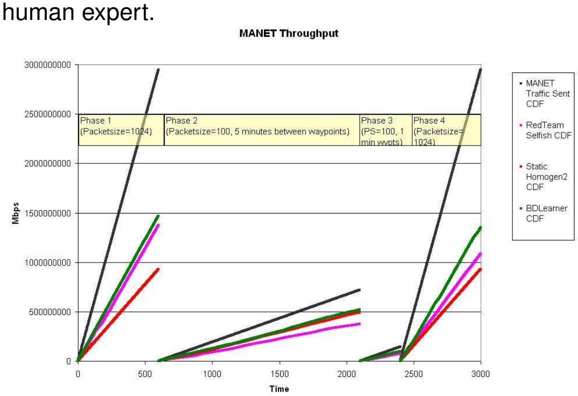

Figure 7. Cumulative MANET throughput for three control approaches.

node j. $\hat{T}_{ij}$ decreases as the distance between the nodes increases.

- P_{j7} is the probability that node j can reach node 7 (stationary command centre). $P_{j7} = 1.0$ if node j is less than 200m from node 7 and drops linearly to 0.0 until node j is farther than 600m from node 7.

Figure 8 shows the distribution of throughput values as calculated by the HDL model and compared to actual throughput, showing that the model is only a rough guide to the ANN.

4.3.1 Scenario A: New Mobility. The training scenarios are as described in Section 4.1.1. In the test scenario, node 1 moved around the ring, while nodes 2 to 7 remained stationary in the upper corner.

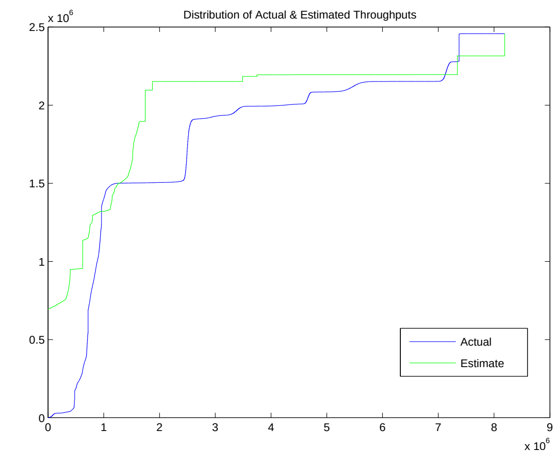

Figure 8. HDL's estimates of throughput $\hat{T}_g$ provide only a very rough estimate of actual throughput T_g.

Throughput	BDL	HDL	Imprvmt
Phase 1	1,192,591,360	1,660,256,256	139%
Phase 2	514,967,808	566,825,728	110%
Phase 3	107,325,600	112,959,200	105%
Phase 4	1,246,363,648	1,604,278,240	129%
Total	3,061,248,416	3,944,319,424	129%

Table 3. HDL outperforms BDL by approximately 30%, showing that HDL more effectively transfers previous experience to a new domain—altered mobility patterns.

Table 3 and Figures 9 and 10 compare the results of using the BDL controller and using the HDL controller. The results show that the hybrid learning approach transfers knowledge more effectively to a new domain. The results are more pronounced in heavier traffic conditions.

4.3.2 Scenario B: New Communications Environment. For the training environment, we used a *FreeSpace* with a line-of-sight closure pathloss model for terrain under normal conditions. For the test environment, we used the *Longley Rice Propagation* pathloss model with these parameters: Surface refractivity at 370; relative permittivity at 7; ground conductivity at 0.002.

Table 4 and Figures 11 and 12 show shows that the hybrid learner outperforms the basic learner when trans-

Figure 9. MANET throughput for HDL and BDL.

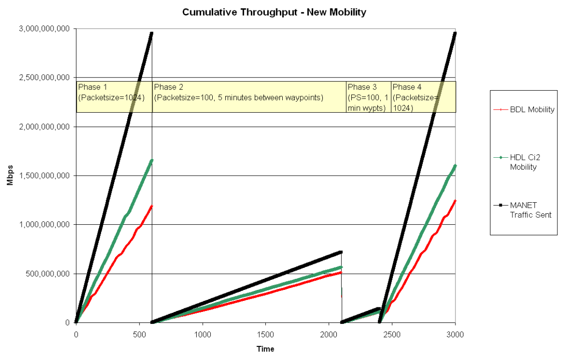

Figure 10. New mobility patterns: Cumulative (by phase) MANET throughput for HDL and BDL.

ferred to this new domain; again, the results are more pronounced for heavier traffic conditions. (In phase 3, BDL outperforms HDL, showing an opportunity for tracking the accuracy of the learned models and dynamically switching among them.) Throughout the mission, HDL successfully transferred 127% of the traffic that the BDL transferred.

5 Conclusions and Future Work

In this paper, we present ORACLE, a distributed learning approach to tuning network configuration parameters for a MANET. We pose the problem formally and describe the techniques used to learn the relationships among configuration parameters and to adaptively optimize mission objectives. We then use simulations to demonstrate the feasibility of ORACLE as well its effectiveness in transferring learned knowledge to new

Throughput	BDL	HDL	Imprvmt
Phase 1	1,192,591,360	1,660,256,256	139%
Phase 2	527,997,536	525,056,928	99%
Phase 3	103,145,600	96,675,200	93%
Phase 4	1,259,855,616	1,627,142,176	129%
Total	3,083,590,112	3,909,130,560	127%

Table 4. HDL outperforms BDL by approximately 30%, effectively transferring knowledge to a new environment.

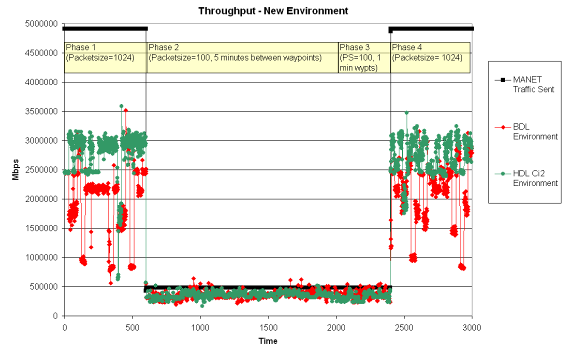

Figure 11. Experiment #2B shows that a hybrid learner transfers previous experience much more effectively to a new domain— altered communications conditions. (a) Throughput

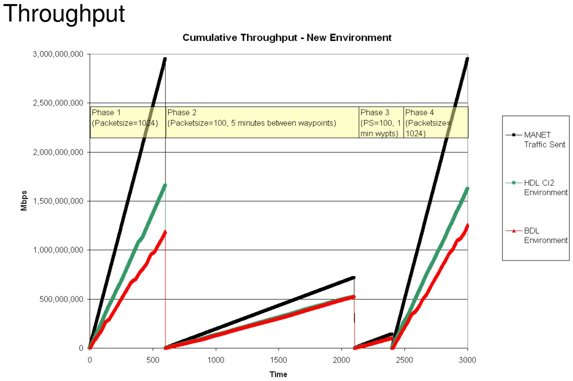

Figure 12. New communications conditions: Cumulative (by phase) MANET throughput for HDL and BDL.

objective functions and environments. Each node in the MANET has its own independently trained controller that observes only *local* conditions but successfully improves the *global* MANET performance.

There are many avenues for further work; some of the more interesting ones are outlined below:

- To update models continuously (rather than offline), performance feedback needs to be distributed correctly to the nodes. We intend to develop a rapid, low-overhead feedback mechanism using distributed averaging techniques [26, 19].
- To learn models more rapidly than neural networks while still maintaining flexibility and accuracy, we intend to develop ensemble methods [7] that rely on locally-trained neural networks and multiple local regressions [10, 14] and select the most effective controller dynamically.
- To increase learning speed while maintaining accuracy, we will reduce the state space by determining parameter significance and sensitivity.
- To improve knowledge transfer results, we will incorporate more analytical models that can be leveraged by the hybrid learner. We will also

track the accuracy of the models and dynamically adjust their trust values.

Acknowledgments: This work was conducted at the Honeywell Labs under IR&D funding. We also thank Jim Freebersyser and Chris Ramming for their enthusiastic encouragement.

6 References

[1] J. A. Boyan and M. L. Littman. Packet routing in dynamically changing networks: A reinforcement learning approach. In *Advances in Neural Information Processing Systems (NIPS)*, pages 671–678. (San Mateo, CA: Morgan Kaufmann), 1994.

[2] W. W. Brown and T. Krout. Future performance expectations for mobile wireless communication networks. In *AFCEA meeting*, San Diego, CA, January 2006.

[3] D. Caprioni and A. Russo. Small unit operations situation awareness system (SUO-SAS) radio architecture and system field testing results. In *Military Communications Conference*, volume 1, pages 198–203. (New York, NY: IEEE Press), 2003.

[4] K. Chen, Y. Xue, and K. Nahrstedt. On setting TCP's congestion window limit in mobile ad hoc networks. In *Proc. IEEE International Conference on Communications (ICC)*, 2003.

[5] Q. Chen and Z. Niu. A delayed adaptive retransmission scheme for false route failure in MANET. In *Proc. 5th International Symposium on Multi-Dimensional Mobile Communications*, pages 858–862, 2004.

[6] C. Chien, M. Srivastava, R. Jain, P. Lettieri, V. Aggarwal, and R. Sternowski. Adaptive radio for multimedia wireless links. *IEEE Journal on Selected Areas in Communications*, 17(5):793–813, May 1999.

[7] T. G. Dietterich. Ensemble methods in machine learning. In J. Kittler and F. Roli (eds), *First International Workshop on Multiple Classifier Systems, Lecture Notes in Computer Science*, volume 1857, pages 1–15. (Berlin, Germany: Springer-Verlag), 2000.

[8] J. Dowling, E. Curran, R. Cunningham, and V. Cahill. Using feedback in collaborative reinforcement learning to adaptively optimize MANET routing. *IEEE Transactions on Systems, Man and Cybernetics*, 35(3):360–372, 2005.

[9] D. A. Eckhardt and P. Steenkiste. Improving wireless LAN performance via adaptive local error control. In *Proc. IEEE International Conference on Network Protocols (ICNP)*, pages 327–338, 1998.

[10] J. Fox. Nonparametric regression (appendix). In *An R and S-PLUS Companion to Applied Regression*. (Thousand Oaks, CA:Sage), January 2002.

[11] E. Gelenbe, R. Lent, and A. Nunez. Self-aware networks and QoS. *Proc. IEEE*, 92(9):1478–1489, Sept. 2004.

[12] H. Gharavi and K. Ban. Cross-layer feedback control for video communications via mobile ad-hoc networks. In *Proc. IEEE 58th Vehicular Technology Conference*, pages 2941–2945, 2003.

[13] T. Goff, J. Moronski, D. S. Phatak, and V. Gupta. Freeze-TCP: A true end-to-end TCP enhancement mechanism for mobile environments. In *Proceedings of the International Conference on Computer Communications INFOCOM*, pages 1537–1545, 2000.

[14] W. L. Gorr, D. Nagin, and J. Szczypula. Comparative study of artificial neural network and statistical models for predicting student grade point averages. *International Journal of Forecasting*, 10:17–34, 1994.

[15] K. Gurney. *An Introduction to Neural Networks*. (London, U.K.: UCL Press), 1997.

[16] G. Holland, N. Vaidya, and P. Bahl. A rate-adaptive MAC protocol for multi-hop wireless networks. In *Seventh Annual ACM International Conference on Mobile Computing and Networking (MOBICOM)*, July 2001.

[17] K. W. Kim, P. Lorenz, and M. M. O. Lee. A new tuning maximum congestion window for improving TCP performance in MANET. In *Proc. Systems Communications*, pages 73–78, 2005.

[18] P. Lettieri and M. B. Srivastava. Adaptive frame length control for improving wireless link throughput, range and energy efficiency. In *Proc. International Conference on Computer Communications INFOCOM (2)*, pages 564–571, 1998.

[19] C. C. Moallemi and B. V. Roy. Consensus propagation. In *Proc. Neural Information Processing Systems*, Vancouver, Canada, December 2005.

[20] D. Montana, T. Hussain, and T. Saxena. Adaptive reconfiguration of data networks using genetic algorithms. In *Proceedings of the Genetic and Evolutionary Computation Conference (GECCO)*, 2002.

[21] A. Parlos, S. Menon, and A. Atiya. An algorithmic approach to adaptive state filtering using recurrent neural networks. *IEEE Transactions Neural Networks*, 12(6):1411–1432, Nov. 2001.

[22] J. C. Ramming. BAA 05-42 Proposer Information Pamphlet. www.darpa.mil/ato/solicit/CBMANET/BAA05-42PIP08.30.05.doc, 2005.

[23] C. Steinbach. A reinforcement-learning approach to power management. Master's thesis, Artificial Intelligence Laboratory, MIT, Cambridge, MA, 2002. Available as AI Technical Report 2002-007.

[24] M. Sugano and M. Murata. Performance improvement of TCP on wireless ad hoc network. In *IEEE Vehicular Technology Conference (VTC)*, Apr. 2003.

[25] N. Tao, J. Baxter, and L. Weaver. A multi-agent, policy-gradient approach to network routing. In *Proc. International Conference on Machine Learning (ICML)*, pages 553–560. (San Mateo, CA: Morgan Kaufmann), 2001.

[26] J. N. Tsitsiklis. *Problems in Decentralized Decision Making and Computation*. PhD thesis, Massachusetts Institute of Technology, Cambridge, MA, 1984.

[27] T. Ye, H. T. Kaur, and S. Kalyanaraman. Large-scale network parameter configuration using an on-line simulation framework. In *IEEE/ACM Transactions of Networking*, 2003.

[28] L. Yin and G. Cao. Adaptive power-aware prefetch in wireless networks. *IEEE Transactions on Wireless Communications*, 3(5):1648–58, Sept. 2004.