

Reinforcement Learning for Routing in Ad Hoc Networks

Petteri Nurmi

Helsinki Institute for Information Technology HIIT
Department of Computer Science, P.O. Box 68, FI-00014 University of Helsinki, Finland
petteri.nurmi@cs.helsinki.fi

Abstract— We show how routing in ad hoc networks can be modeled as a sequential decision making problem with incomplete information. More precisely, we show how to map routing into a reinforcement learning problem involving a partially observable Markov decision process, and present an algorithm for optimizing the performance of the nodes in this model. We also present simulation results with our model.

I. INTRODUCTION

In communication networks, routing refers to the problem of finding the best path through which to send packets bound for a given destination. The standard solution for routing is to consider the network as a weighted graph and to find the path with the minimum cost in this graph. The weights of the edges are set according to some cost criterion that can consider, e.g., latency, link failures, congestion, and selfishness. However, once we move to decentralized settings, such as ad hoc networks, the optimization needs to be performed locally on each device and the cost of an edge depends on how each node perceives the performance it obtains by using the edge. In ad hoc networks an additional factor that can impact the performance of the nodes, and thus the costs of the edges, is resource constraints and especially energy [1].

In the literature, reinforcement learning has been suggested as a method for estimating the cost of an edge under changing conditions, and using only local information (e.g., [2]–[5]). Existing solutions have been targeted at static communication networks where energy is not an issue. Therefore we need to modify the existing solutions before they can be used in ad hoc networks. In this paper we propose a novel approach that combines reinforcement learning with online parameter estimation so that nodes learn stochastic policies that map beliefs about local properties of a neighboring node to an estimated cost for an edge. As our first contribution, we formulate this problem as a partially observable Markov decision process (POMDP). As our second contribution, we present an algorithm that allows the nodes to optimize their routing behavior in this model. The algorithm combines stochastic approximation and function approximation to learn a stochastic controller whose output depends on the values of the parameter estimates. Finally, we use simulations to give insights into the expected behavior of the nodes.

The rest of the paper is organized as follows: Sec. II introduces related work. Sec. III formalizes routing in ad hoc

networks as a decision problem under incomplete information. Sec. IV presents an algorithm for optimizing the routing behavior of a node and Sec. V presents experimental results. Finally, Sec. VI concludes the paper.

II. RELATED WORK

A. Modeling Routing

Although game theory has only recently been applied to ad hoc networks, other types of communication networks have been analyzed using game theoretic tools. Especially game theoretic analysis of Internet congestion control has been an active research area [6]–[9]. However, due to node mobility and resource constraints, ad hoc networks have more inherent sources of uncertainty than other types of communication networks. As a consequence, models that are designed specifically for ad hoc networks are needed. In this section we survey related work on modeling routing behavior in ad hoc networks.

Srinivasan et al. [10] model routing using the repeated prisoners' dilemma (see, e.g., [11]). In this model, a node has two possible modes of behavior: to cooperate (forward packets) or to defect (drop packets). The authors propose a strategy called generous tit-for-tat (GTFT), according to which a node i forwards packets for node j with a probability that depends on how many packets node j has forwarded for node i . The authors also prove that GTFT is Pareto optimal, i.e., it allocates the resources of the nodes optimally. Our model is more generic as it takes into account also other aspects than selfishness. In addition, our model takes into account uncertainty about the resources of other nodes.

Altman et al. [12] assume that each node j forwards packets with a probability μ_j that is independent of the source of traffic. When a packet is sent to the network, each node computes the current (mixed) equilibrium strategy and uses it as the forwarding probability. The authors also propose a punishment mechanism where nodes decrease their forwarding probabilities, when someone deviates from the equilibrium strategy. The authors prove that the probability given by the equilibrium strategy increases with the number of intermediate nodes. Furthermore, the authors give an algorithm for calculating the equilibrium strategies in a distributed manner. The main problem of this approach is that the routing model is not generic as it is unlikely that the forwarding probabilities of the nodes are independent of the source. In addition,

the complexity of calculating the equilibrium strategy is not feasible in terms of storage and computational effort. Finally, also this approach ignores the resources of the devices and the uncertainty about the resources of other nodes.

Urpi et al. [13] use static Bayesian games and infinitely repeated games to model routing. In their model nodes are divided into classes so that nodes in the same class generate traffic according to the same stochastic process. Once assigned, the energy class of a node stays fixed for the entire lifetime of the node. A node can only have information about nodes that belong to its neighborhood Γ_i . Nodes in the neighborhood of node i are assumed to have beliefs about its energy class and vice versa. In each repetition of the game a node has to decide how many packets to send to the network and how many packets to forward for neighboring nodes. The model has the same problems as the others we have discussed. The uncertainty about resources of others is ignored, and the model utilizes only local information, which implies that no guarantees can be given that packets actually reach the correct destination. Another problem with the model is that the players' information increases over time and hence the beliefs of the players should be updated. However, in the model the beliefs remain the same.

B. Mechanisms Against Selfishness

Mechanisms for coping with misbehaving nodes can be categorized into *incentive* based and *detection and exclusion* based. Incentive based mechanisms attempt to make cooperation attractive by giving nodes monetary benefits when they forward packets for others. Incentive based mechanisms include all virtual currency systems, such as Nuglets [14], Sprite [15], ad hoc-VCG [16], CASHnet [17] and Secure Incentive Protocol (SIP) [18]. Virtual currency systems suffer from various problems: centralized approaches suffer from the need of trusted third-party entities whereas decentralized approaches usually cause significant overhead or require the existence of tamper-free hardware. In addition, nodes on the borders of the network may run out of currency as nobody requires them to forward packets.

The detection and exclusion based mechanisms, on the other hand, attempt to detect misbehaviors and use a distributed reputation system to exclude, at least temporarily, nodes that act selfishly. The *Watchdog* is generally considered to be the first reputation related approach for ad hoc networks [19]. The idea in the Watchdog mechanism is to use overhearing to monitor traffic from neighboring nodes and to detect misbehaving nodes. A *PathRater* module is then used to rate routes and to avoid routes with misbehaving nodes. The main problem with the Watchdog is that it does not penalize selfishness, but rather encourages it. Namely, no messages are routed through the misbehaving nodes, but their traffic is still being forwarded. Other distributed reputation systems include Core [20], OCEAN [21], CONFIDANT [22], SORI [23] and the GTFT mechanism [10]. The approach that we use in this paper is closely related to the CONFIDANT system.

C. Energy-Aware Routing

Also the problem of routing in an energy efficient way has received much attention. Initially, the focus was on developing routing protocols, which allow finding the path that minimizes energy consumption [24]–[26]. As noted in [27], if all traffic is routed through the minimum energy path, energy consumption is unbalanced and nodes start dying out sooner. As a consequence, solutions that attempt to balance energy usage have been proposed. For example, [28] proposes a scheme where nodes maintain multiple paths to the destination and use a probabilistic model that depends on an energy metric to select which path to use. As another example, [27], [29] formulates energy efficient routing as a linear programming (LP) problem with additional capacity constraints. The solutions to the LP provide an approximation to the optimal traffic allocation in the network.

Contrary to our work, all of the proposed approaches assume that the willingness of a node to forward packets does not depend on its energy. However, simulation results in [1] show that modifying routing behavior on the basis of remaining energy increases the expected lifetime of a node as its resources are used in a more balanced fashion. As a consequence, this should be the rational choice for the nodes. Hence our work supplements existing approaches by considering also the effect of energy on routing decisions.

III. PROBLEM FORMULATION

In our model, the decision problem that node i faces is to select which node to use as the next hop, under the assumption that routing decisions of intermediate nodes are stochastic and depend on some private information. The main question to answer is how can nodes optimize their performance? In order to be able to answer this question properly, this section formalizes the decision problem node i is facing, and shows that the decision problem can be mapped to a POMDP. In the next section we present an algorithm that node i can use to optimize its performance.

A. Single Stage

Let $N = \{1, \dots, n\}$ denote the set of nodes in an ad hoc network. We assume the network to be connected, i.e., that every node can reach every other node in the network. We assume that each node $i \in N$ has a discrete representation for time so that a new time step t begins when the agent has new packets to be sent to the network. Each time step t corresponds to a new *stage* or *epoch* of the decision problem that the agent is facing.

We consider routing from the point of view of an individual node i and assume that at time t it wants to send packets to some node d . For notational simplicity, we assume the destination to be same for all packets that are sent at time t . Although we restrict ourselves to the point of view of node i , in practice all nodes would behave similarly as node i when they want to send packets to the network. Furthermore, any number of nodes can send packets to the network simultaneously.

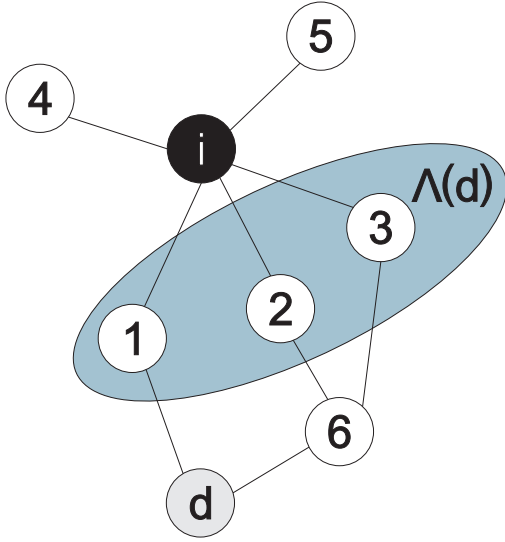


Fig. 1. An illustration of the concept of one-hop neighborhood. In the example, the nodes $\{1, 2, 3\}$ form the one-hop neighborhood $\Lambda(d)$ because the routing protocol thinks packet can be routed to d using one of them as the first hop.

We say that the destination d is *reachable* from node $j \in N \setminus \{i\}$, if, according to the used routing protocol, there is a route from i to d that goes through j . Here we assume that the routing protocol is able to return also paths that are relatively close to optimal. In our experiments we use a modified version of DSR. We define

$$\Lambda(d) := \{j | j \in N \setminus \{i\} \text{ and } d \text{ is reachable from } j\} \quad (1)$$

as the *one-hop neighborhood* of node i for destination d . The concept of one-hop neighborhood is illustrated in Fig.1. We do not make any assumptions regarding the routing protocol that is used. The advantage of this is that our model is applicable both to the case where network links are directed and to the case where they are undirected.

Let $g(t)$ denote the number of packets node i wants to sent to the network at time t . The stage t action of node i , denoted $a_i(t)$, is a vector that indicates how many packets are sent to each of the nodes in $\Lambda(d)$. The vector $a_i(t)$ lies within a simplex spanned by the vectors $g(t)e_j$, where e_j is the unit vector along the j :th coordinate axis.

We use $a_j(t)$ to denote the action of the intermediate node $j \in \Lambda(d)$, i.e., how many packets node j forwarded of node i 's stage t traffic. If the packets are not sent through j , we define $a_j(t) = 0$ to be the empty action, i.e., node j did nothing. We use $a_{-i}(t)$ to denote the *deleted* action profile that contains the (stage t) actions of all nodes except i , i.e.,

$$a_{-i}(t) = (a_1(t), \dots, a_{i-1}(t), a_{i+1}(t), \dots, a_n(t)). \quad (2)$$

Furthermore, we define the stage t action history of node j , denoted \mathcal{H}_j^t , as the collection

$$\mathcal{H}_j^t = (a_j(1), \dots, a_j(t-1)). \quad (3)$$

We also use \mathcal{H}^t to denote the collection of stage t action histories over the nodes in $N \setminus \{i\}$.

The decision of an intermediate node j can depend on various factors, for example, on the current congestion level, the reputation of node i , the selfishness of node j and the remaining energy of node j . In practice it is impossible to model all of these factors exactly, and we assume instead that the behavior of node j is stochastic and it depends on a finite number of parameters that are independent and identically distributed (i.i.d.), and unknown to node i . We use σ_j to denote the forwarding probability of node j . Furthermore, we use θ_j^l ($l = 1, 2, \dots, m$) to denote the parameters and $\theta_j^l(t)$ to denote the value of parameter θ_j^l at time t . Without loss of generality, we assume that the values of the parameters lie in the interval $[0, 1]$. For notational simplicity, we use $\Theta_j(t)$ to denote collectively $\theta_j^l(t)$ for all l , i.e.,

$$\Theta_j(t) = (\theta_j^1(t), \dots, \theta_j^m(t)). \quad (4)$$

Thus, we have assumed that the forwarding probability of node j is a function that depends on the current values of the parameters $\Theta_j(t)$, i.e.,

$$\sigma_j = G(\Theta_j(t)), \quad (5)$$

where G is some suitable function¹. In later sections we study two special cases of our model: one where the decisions of the nodes depend only on their selfishness and one where their decisions depend also on their remaining energy. We reserve θ_j^1 for denoting the selfishness of node j and θ_j^2 for denoting its remaining energy level.

When node i is deciding which node to use as the next hop, it requires information about $\Theta_j(t)$ to be able to make an optimal decision. Accordingly, at time t , node i faces a decision problem, whose *state* is given by the vector

$$\mathbf{s}(t) = (\Theta_1(t), \dots, \Theta_{i-1}(t), \Theta_{i+1}(t), \dots, \Theta_n(t)). \quad (6)$$

Unfortunately the state $\mathbf{s}(t)$ is not usually directly observable. Namely, if the values of the parameters affect the behavior of an intermediate node, it has no incentive to reveal this information to others.

Although the values of the variables are unknown to node i , it can derive estimates about the values from past experiences with node j . Accordingly, we need to require that node i can reliably observe the actions of node j ; see, e.g., [19], [30] for details about how to obtain this information.

We use $\hat{\Theta}_j(t)$ to denote node i 's estimate of $\Theta_j(t)$. To be precise, we maintain for each parameter θ_j^l a full probability distribution over $[0, 1]$. However, we assume that this distribution is fully characterized by its mode $\hat{\theta}_j^l(t)$, which we then try to estimate. Respectively, we use $\hat{\mathbf{s}}(t)$ to denote the vector

$$\hat{\mathbf{s}}(t) = (\hat{\Theta}_1(t), \dots, \hat{\Theta}_{i-1}(t), \hat{\Theta}_{i+1}(t), \dots, \hat{\Theta}_n(t)). \quad (7)$$

¹With suitable, we mean here that the probabilities reflect the true preferences of the node j . For example, that the value of G decreases as the energy of j decreases or as the estimate of the selfishness of node i increases.

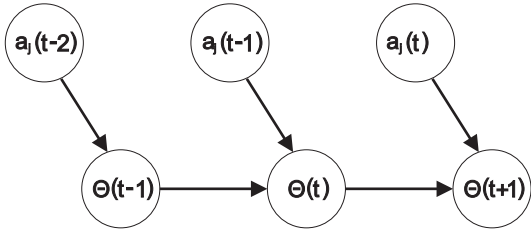


Fig. 2. A graphical model illustrating the dependencies between the parameters and the actions of agent j .

By definition, the estimates $\hat{\Theta}_j(t)$ are functions of the corresponding action histories \mathcal{H}_j^t . If we need to consider the entire action history, calculating these estimates easily becomes infeasible. Therefore, we make some assumptions about the parameters Θ_j and the decision making of the intermediate nodes j .

First of all, we assume that the action of node j depends only on the current values of the parameters, i.e., on $\Theta_j(t)$. Next, we assume that the value of $\Theta_j(t)$ depends only on the previous value $\Theta_j(t-1)$ and on the previous action $a_j(t-1)$. In other words, we have assumed that the parameters have the *Markov property*, i.e., for all l , $\theta_j^l(t)$ follows a Markov chain. These dependencies are also illustrated in Fig. 2. The assumptions mean that the estimate $\hat{\Theta}_j(t)$ needs to consider only the previous action of node j and the previous distribution of the estimate.

B. Overall Model

The system model that we considered in the previous section specifies the stage t decision problem of node i . In order to finalize the specification of our model, in this section we consider the overall model that results when we look at all of the stages over time. To this end, we give a detailed specification of the components of the resulting POMDP. The section is mainly of theoretical interest and a reader who is more interested in practical details can skip the section.

Because of, e.g., link failures, node movements and different destinations, the nodes with which node i interacts vary over time. When we look at all interactions over time, node i can potentially send packets through every other node in the network. Let g_{\max} be an upper bound on the number of packets a node can process during a single time step. The overall action space of the decision problem, denoted \mathcal{A} , then is a $n-1$ dimensional simplex, spanned by the vectors $g_{\max} e_j$.

According to the same argument, also the overall state space of the decision problem, denoted \mathcal{S} , must consider all nodes in the network. Since we assumed that the decisions of the intermediate nodes depend on m parameters whose values lie in the interval $[0, 1]$, the overall state space is the $m \times (n-1)$ dimensional unit cube.

When node i is making its stage t decision, the information that is available to it consists of the stage t action histories, i.e., of \mathcal{H}^t . Thus, clearly the space of available information is given by $\{\mathcal{H}^t\}_{t=0}^{\infty}$. Following common reinforcement learning terminology, this is called the *observation space* of the

decision problem.

We also assume that the agents have a real valued reward function $r : \mathcal{A} \times \mathcal{S} \rightarrow \mathbb{R}$. The reward function is related to the utility agent i gains when its packets are forwarded to the destination or when they are dropped². We assume the function r to be of von Neumann-Morgenstern type, i.e., that it preserves the underlying preference structure of node i .

The final component of the decision problem that we must specify are the state transitions, i.e., how the values of $\theta_j^l(t)$ change over time. We discuss the state transitions more in Sec. IV and here we merely assume that the transitions are given by some, potentially unknown, distribution G that satisfies the Markov property³.

We can now formulate the decision problem as a *partially observable Markov decision process* (POMDP) as follows:

- the *action space* is \mathcal{A}
- the *state space* is \mathcal{S}
- the *observation space* is $\{\mathcal{H}^t\}_{t=0}^{\infty}$
- the *rewards* are given by function r
- the *state transitions* are given by some, potentially unknown, distribution G .

The Markov assumption implies that nodes can concentrate on optimizing their performance from time t onward instead of considering the entire history of interactions. Accordingly, at stage t , the decision problem of node i is fully characterized by the tuple $\langle \mathcal{A}(t), \mathcal{S}(t), \hat{\mathcal{S}}(t), \mathcal{H}^t \rangle$. This implies that developing an algorithm for optimizing the performance of the nodes requires us to (i) have a method for computing the estimates $\hat{\mathcal{S}}(t)$ and (ii) have a method for computing the optimal action given the available information. In Sec. IV we show how to do this in practice.

C. When Optimizing the Next Hop Suffices?

In the previous section, we have formulated routing as a reinforcement learning problem with incomplete information. In our formulation, nodes estimate the "cost" of links over time and use the link with the cheapest cost for routing.

The single hop case does not guarantee that packets are actually routed to the destination. If we generalize our model to the multi-hop case, this means that node i must optimize the entire routing path, which significantly increases the complexity of the model. In order to avoid the increased complexity, in this section we give conditions under which optimizing only the next hop suffices.

We assume that for each intermediate node j the set $\Lambda(d)$ is such that the resulting path is close to the globally optimal route from the point of view of the source node i , i.e., that the route does not have loops and that it eventually reaches the destination. This assumption is valid, e.g., when the packet headers identify which nodes have already seen the packet and

²More precisely, the reward corresponds to the marginal utility of player i , i.e., $r(a(t); s(t)) = u_i(a_i(t); a_{-i}(t); s(t))$.

³The assumptions we have made can be understood so that the behavior of a node follows a multinomial distribution whose parameters change over time according to the state transition distribution G .

the routing protocol ensures that the packets are not sent to these nodes anymore.

Intuitively, optimizing only the next hop suffices if a node can be guaranteed that also the next hop will do the same. On the other hand, we can be ensured that the next hop is willing to optimize the subsequent hop only if it has an incentive to do so. Note that while this scheme does not ensure a globally optimal solution, it attempts to give incentives to make the locally optimal path attractive.

A sufficient incentive is to require that the destination node acknowledges the packets. In this scheme intermediate nodes are rewarded only when the packets have arrived at the correct destination. On the other hand, we should not punish nodes if they make an effort to optimize the route of the packets. When some intermediate node drops the packets, the previous hop should punish the node that dropped the packets. This node, however, should not be punished as it has made an effort to optimize the route. But it also should not be rewarded since the packets were not routed to the destination. We are currently working on an extended neighborhood watch mechanism that is based on these principles.

IV. AN ALGORITHM FOR LEARNING OPTIMAL STRATEGIES

In POMDPs the agents are usually assumed to find a policy π that optimizes their long term utility. A typical measure for the utility is to consider the discounted infinite horizon sum

$$V^\pi(t) = \sum_{k=t}^{\infty} \delta^{k-t} r(a_i(k); \mathbf{s}(k)), \quad (8)$$

where $r(a_i(k); \mathbf{s}(k))$ is the reward of agent i for executing action $a_i(k)$ in state $\mathbf{s}(k)$ and δ is a discount factor. Maximizing this quantity, however, requires information about the environment dynamics. Alternatively we can try to learn a model of the environment dynamics and to use the estimated dynamics for calculating the utility [31]. The third alternative is to set the discount factor to zero and to construct rules that ensure that the learning converges to a state that is optimal in some sense [32].

In this paper we follow the third approach, in which case the decision problem reduces to a *multi-arm bandit* problem. The arms of the bandit correspond to neighboring nodes $j \in \Gamma$ and the reward from arm j is a function of unobservable, arm specific parameters.

Because the overall state space is infinite dimensional, we cannot learn an exact policy. Instead we must resort to approximation techniques. A widely used technique for learning an approximate policy is to use function approximation techniques (e.g., [33], [34]) to learn an approximate controller, i.e., learn a control function $f_j(\Theta_j(t))$ that approximates the optimal policy σ_j as well as possible. For notational simplicity, we assume that all nodes behave similarly, which implies that we can write $f_j = f$ for all j . The algorithm that we present can be used also when this assumption is not valid.

We require the function $f(\cdot)$ to be differentiable and strictly increasing in Θ_j . The latter assumption follows directly from

the assumptions we made regarding the behavior of the intermediate nodes. The former assumption, on the other hand, ensures that we can optimize function f efficiently. In practice requiring differentiability is not a problem. Some examples of possible functional forms for f include the hyperbolic tangent function, the sigmoid function and the weighted linear sum function, i.e., functions that are used as activation functions in neural network literature [35].

When the function $f(\cdot)$ is differentiable, we can optimize it using stochastic gradient descent. Let $E(\cdot)$ be a loss function that measures the error between the observed and estimated probability. We require also E to be differentiable. Define β to be the ratio between the number of packets forwarded by node j and the number of packets sent to node j , i.e., $\beta = \text{packets forwarded} / \text{packets sent}$. Let $f^t(\cdot)$ be the current estimate of the controller. In the experiments, we use $f^t(\Theta_j(t)) = \mathbf{w} \cdot \Theta_j(t)$ as the control function. Here \mathbf{w} is a real-valued weight vector. As the loss function we use $E = 0.5(f^t(\Theta_j) - \beta)^2$. The stochastic gradient algorithm updates the estimate of f , denoted by $f^t(\cdot)$, following the direction of the gradient of the loss function, i.e.,

$$f^{t+1} = f^t + \eta \frac{\partial E}{\partial \Theta_j}, \quad (9)$$

where η is the step size of the gradient. We return to the selection of step size η later in the section.

In addition to learning a controller, the agents need to be able to estimate Θ_j . In our case, we use a variable learning rate stochastic approximation algorithm that attempts to track the exact values from noisy observations. While it has been shown that the variance of stochastic approximation methods can in general be large [36], especially when no discounting (as in our case) is used, they also have been shown to be efficient in multi-agent decision making [37].

The scheme we use is an adaptation of the win-or-lose-fast policy hill climbing algorithm (WoLF-PHC) [37], [38]. In WoLF-PHC learning is governed by two constants α_w and α_l , for which $0 < \alpha_w < \alpha_l \leq 1$ holds. In the experiments we use the values $\alpha_w = 0.15$ and $\alpha_l = 0.30$. Whenever we are "winning", the constant α_w is used for the updates whereas if we are "losing" the constant α_l is used. The motivation for this is that when we are winning, also other agents are updating their strategies, which causes the environment to be non-stationary. Usually the learning processes converges eventually to a stationary distribution, in which case we can be ensured of the convergence of the policy climbing algorithm, if the agents are "patient" enough in their updates [37], [39].

Whenever the source node needs to make a decision, it calculates the value of the controller for all nodes in the set $\Lambda(d)$ given the current estimates of energy and selfishness for the nodes. It then selects the greedy action, i.e., the node that is most likely to forward the packets, with probability $1 - \epsilon_t$ and a randomly selected node, that does not equal the greedy choice, with probability ϵ_t . Thus, we require that node i performs exploration with the possible actions; see [33] for possible exploration rules. In this paper we use the rule $\epsilon_t = 1/t$.

A. Estimating Energy and Selfishness

Before giving the exact update formulas for energy and selfishness estimates, we have to define what is meant by winning and losing in our case. Let $\hat{\Theta}_j(t)$ be our current estimate. First of all, we use an additional estimate $\rho_j(t)$ to track the average rate at which node j sends packets. This estimate is calculated using

$$\rho_j(t+1) = \rho_j(t) + \eta' (\beta - \rho_j(t)), \quad (10)$$

where η' is the step size. In the experiments we use 0.15 ($= \alpha_w$) as the step size. This estimate is used to define winning for the energy estimate so that the update equation for $\hat{\psi}_j(t)$ becomes

$$\hat{\theta}_j^2(t+1) = \begin{cases} \hat{\theta}_j^2(t) + \alpha_w (1.0 - \hat{\theta}_j^2(t)) & \beta > \rho_j(t) \\ \hat{\theta}_j^2(t) + \alpha_l (0.0 - \hat{\theta}_j^2(t)) & \text{otherwise.} \end{cases} \quad (11)$$

The value of the estimator usually decreases. The motivation for this is that we attempt to model the decrease in energy over time. When the rate of forwarding exceeds the average rate, then our assumption is that energy has increased and we attempt to update the estimate upward. This estimator is somewhat crude and in practice we should use, e.g., information about the packets sent by j to derive a more accurate estimator. In our case, we use this estimator simply to give an idea of the expected behavior of the nodes. The same rule for winning is also used for the step size η in Eq. 9 so that $\eta = \alpha_w$ when $\beta > \rho_j(t)$ and $\eta = \alpha_l$ otherwise.

For the estimation of selfishness, we use the following equations

$$\hat{\theta}_j^1(t+1) = \begin{cases} \hat{\theta}_j^1(t) + \alpha_w (\beta - \hat{\theta}_j^1(t)) & \beta > \hat{\theta}_j^1(t) \\ \hat{\theta}_j^1(t) + \alpha_l (\beta - \hat{\theta}_j^1(t)) & \text{otherwise.} \end{cases} \quad (12)$$

The justification for this estimators comes from game theory and we refer to [40] for more details.

V. EXPERIMENTS

In order to give insights into node performance under our model, we have conducted a set of simulation experiments. The goal of the experiments is to show how an energy constrained network might function and to highlight what kind of effects we can expect to see in the network. In the following subsections we first detail our simulation setup, after which we present results from the simulations.

A. Simulation setup

In our simulations, a single experiment consisted of 1000 iterations (seconds). The number of nodes was fixed (100) and the topology of the network was randomly generated. Each experiment setup was repeated 30 times. In each experiment, we assigned a random amount of nodes as selfish. The number of selfish nodes was drawn using a Bernoulli process so that the average amount of selfish nodes was 10. When a node was

assigned selfish, the value of θ_j^1 was drawn randomly from the interval $[0.8, 1.0]$.

The nodes were deployed into a 125 m \times 125 m area and the range of the nodes' transmitters was set to 25 meters. In the initial deployment we required that the minimum distance between two nodes is 10 meters. This way the network topologies were initially fairly uniform.

The movements of the nodes were modeled using a random waypoint model with constant speed (4 m/s) and Poisson distributed waiting times. The mean of the waiting times was set to 10 seconds ($=$ iterations). A Poisson distribution was used also for traffic generation. The mean of the traffic generating distribution was set to five and hence the nodes would generate about 500 packets per iteration.

Energy consumption was modeled using a squared loss function, according to which the energy C that is needed to send a packet is given by

$$C = \lambda d^2, \quad (13)$$

where d is the distance over which the packets are sent and λ is a scaling parameter (see Table I). The coefficient and the energy of the nodes was set so that first nodes would start to die out approximately at around 100 iterations.

An important factor in simulating our model is to specify the utility functions of the nodes. Let $a_{ij}(t)$ denote the number of packets node i sends through node j at time t . In the experiments, we use the function

$$\begin{aligned} u_i(a_i(t)) &= \sum_j u_{ij}(a_{ij}(t)) \\ &= \sum_j f^t(\hat{\Theta}_j(t)) (\alpha \cdot a_{ik}(t) - \exp(a_{ik}(t) \cdot C)), \end{aligned} \quad (14)$$

where α is a predefined constants and C is the energy that is needed to send the packets (as given by Eq. 13).

We modeled the behavior of the intermediate nodes as follows: with probability θ_j^2 the intermediate node drops all packets and otherwise it forwards packets with probability

$$\sigma_j = \exp\left(-\frac{(1.0 - \theta_j^2)}{\nu}\right), \quad (15)$$

where ν is a constant. The former case corresponds to node j being selfish and the latter case to when node j optimizes its behavior according to its remaining energy. We do not claim that our selection of utility functions is realistic, but we merely want to give insights into the performance of the nodes when the utility functions are more complex and non-linear. A summary of the simulation parameters is given in Table I.

B. Results

In order to give an idea of the performance of the nodes, in Fig. 3 we show the success rate of the nodes when the importance of energy is varied. The topmost plot represents the case when nodes do not consider their energy in their decision making whereas the lowest plot corresponds to the case where

Parameter	Value
θ_j^1	[0.8, 1.0]
α	12.5
λ	0.0008525
ν	1.15
Area	125 m \times 125 m
Energy consumption	$C = \lambda d^2$
Initial energy	\sim Uniform(8000, 9000)
Iterations	1000
Movement	random waypoint
Number of nodes	100
Range	25 m
Repeats	30
Speed	constant (4 m/s)
Traffic generation	\sim Poisson(5)
Waiting times	\sim Poisson(10)

TABLE I

SUMMARY OF THE USED SIMULATION PARAMETERS.

they consider only their energy. As the plots indicate, the performance of the nodes drops rapidly, if the nodes consider also their energy. On the other hand, when they consider the energy, the expected lifetime of the nodes becomes longer, which can be seen from Fig. 4. To understand the decrease in performance in the case where nodes do not consider their energy, note that cooperative nodes are more likely to run out of battery than selfish nodes. As a consequence, the relative amount of selfish nodes actually increases over time.

VI. CONCLUSIONS

In this paper we have developed a generic model of routing under incomplete information in ad hoc networks. In our model, the routing situation is mapped into a partially observable Markov decision process, which allows us to apply existing theoretical results and algorithms from the reinforcement learning literature. As a concrete examples of our model, we have discussed a model where the routing decisions of the nodes depend only on their selfishness and a model where the decisions depend also on the energy level of the nodes. Furthermore, we have presented an algorithm that uses stochastic approximation to optimize the behavior of the nodes. Our algorithm generalizes also to models where the decisions depend on an arbitrary (discrete) amount of parameters. Finally, we have presented simulation results that show the performance of the nodes and that give insights into the expected performance of the nodes under our model.

VII. ACKNOWLEDGMENTS

The author is greatly indebted to Huizhen Yu and Jukka Suomela for significantly helping to clarify the ideas presented in this paper. The author is also grateful to professor Dimitri Bertsekas for useful discussions on the topic. This work was supported in part by the IST Programme of the European Community, under the PASCAL network of excellence, IST-2002-506778. The publication only reflects the authors' views.

REFERENCES

- [1] P. Nurmi, "Modeling energy constrained routing in ad hoc networks," in *Proceedings of the Workshop on Game Theory for Networks (GameNets)*. ACM Press, 2006.
- [2] J. A. Boyan and M. L. Littman, "Packet routing in dynamically changing networks: A reinforcement learning approach," in *Advances in Neural Information Processing Systems*, J. D. Cowan, G. Tesaro, and J. Al-spector, Eds., vol. 6. Morgan Kaufmann Publishers, 1994, pp. 671–678.
- [3] S. P. M. Choi and D.-Y. Yeung, "Predictive q-routing: A memory-based reinforcement learning approach to adaptive traffic control," in *Advances in Neural Information Processing Systems*, D. S. Touretzky, M. Mozer, and M. E. Hasselmo, Eds., vol. 8. MIT Press, 1996, pp. 945–951.
- [4] 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 - Part A: Systems and Humans*, vol. 35, no. 3, pp. 360 – 372, May 2005.
- [5] P. Wang and T. Wang, "Adaptive routing for sensor networks using reinforcement learning," in *Proceedings of the 6th IEEE International Conference on Computer and Information Technology (CIT)*. IEEE Computer Society, 2006.
- [6] R. Garg, A. Kamra, and V. Khurana, "A game-theoretic approach towards congestion control in communication networks," *ACM SIGCOMM Computer Communications Review*, vol. 32, no. 3, pp. 47–61, 2002.
- [7] C. Papadimitriou, "Algorithms, games, and the internet," in *Proceedings of the thirty-third annual ACM symposium on Theory of computing*. ACM Press, 2001, pp. 749–753.
- [8] T. Roughgarden, "Selfish routing," Ph.D. dissertation, Cornell University, May 2002.
- [9] S. Shenker, "Making greed work in networks: a game-theoretic analysis of switch service disciplines," *ACM/IEEE Transactions on Networking*, vol. 3, no. 6, pp. 819–831, December 1995.
- [10] V. Srinivasan, P. Nuggehalli, C.-F. Chiasserini, and R. R. Rao, "Cooperation in wireless ad hoc networks," in *Proceedings of the 22nd Annual Joint Conference of the IEEE Computer and Communications Societies (INFOCOM)*. IEEE Computer Society, 2003, pp. 808–817.
- [11] R. Axelrod, *The Evolution of Cooperation*. Basic Books, 1984.
- [12] E. Altman, A. A. Kherani, P. Michiardi, and R. Molva, "Non-cooperative forwarding in ad-hoc networks," in *Proceedings of the 15th IEEE International Symposium On Personal, Indoor and Mobile Radio Communications*, 2004.
- [13] A. Urpi, M. Bonuccelli, and S. Giordano, "Modelling cooperation in mobile ad hoc networks: a formal description of selfishness," in *Proceedings of Modelling and Optimization in Mobile, Ad Hoc and Wireless Networks (WiOpt)*, March 2003.
- [14] L. Buttyán and J.-P. Hubaux, "Stimulating cooperation in self-organizing mobile ad hoc networks," *Mobile Networks and Applications (MONET)*, vol. 8, no. 5, pp. 579–592, 2003.
- [15] S. Zhong, J. Chen, and Y. R. Yang, "Sprite: A simple, cheat-proof, credit-based system for mobile ad-hoc networks," in *Proceedings of the 22nd Annual Joint Conference of the IEEE Computer and Communications Societies (INFOCOM)*. IEEE Computer Society, 2003.
- [16] L. Anderegg and S. Eidenbenz, "Ad hoc-VCG: a truthful and cost-efficient routing protocol for mobile ad hoc networks with selfish agents," in *Proceedings of the 9th annual international conference on Mobile computing and networking (MobiCom'03)*, D. B. Johnson, A. D. Joseph, and N. H. Vaidya, Eds. ACM, 2003, pp. 245–259.
- [17] A. Weyland and T. Braun, "Cooperation and accounting strategy for multi-hop cellular networks," in *Proceedings of the 13th IEEE Workshop on Local and Metropolitan Area Networks (LANMAN)*. IEEE Computer Society, 2004, pp. 193 – 198.
- [18] Y. Zhang, W. Lou, and Y. Fang, "SIP: a secure incentive protocol against selfishness in mobile ad hoc networks," in *Proceedings of the Wireless Communications and Networking Conference (WCNC)*, vol. 3. IEEE Computer Society, 2004, pp. 1679 – 1684.
- [19] S. Marti, T. J. Giuli, K. Lai, and M. Baker, "Mitigating routing misbehavior in mobile ad hoc networks," in *Proceedings of the 6th annual international conference on Mobile computing and networking (MobiCom)*. ACM Press, 2000, pp. 255–265.
- [20] P. Michiardi and R. Molva, "Core: a collaborative reputation mechanism to enforce node cooperation in mobile ad hoc networks," in *Proceedings of the IFIP TC6/TC11 Sixth Joint Working Conference on Communications and Multimedia Security*, ser. IFIP Conference Proceedings,

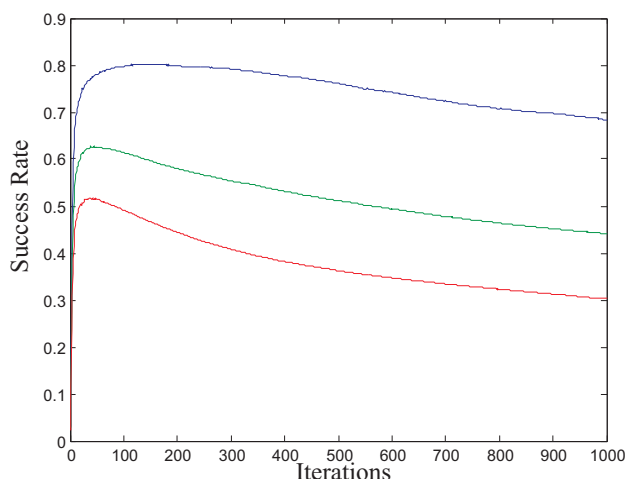


Fig. 3. The (cumulative) success rate of nodes (y-axis) as a function of iterations (x-axis). The plots correspond to the cases where energy is not considered (top), when energy and selfishness are considered equally and when only energy is considered (bottom).

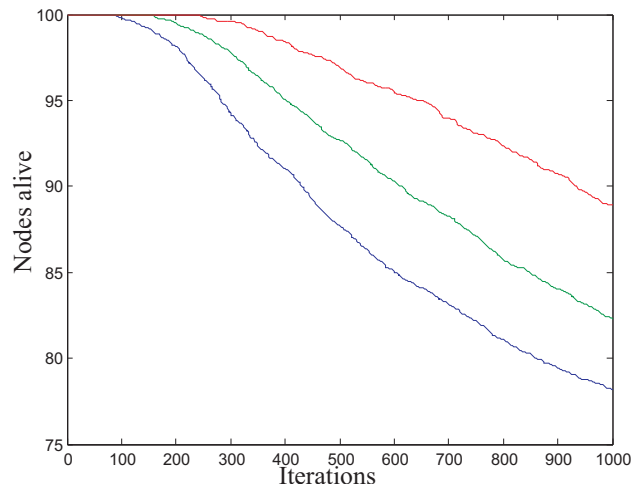


Fig. 4. The amount of nodes that are alive (y-axis) as a function of iterations (x-axis). The plots correspond to the cases where energy is not considered (fastest decrease), when energy and selfishness are considered equally and when only energy is considered (slowest decrease).

- B. Jerman-Blazic and T. Klobucar, Eds., vol. 228. Kluwer, 2002, pp. 107–121.
- [21] S. Bansal and M. Baker, “Observation-based cooperation enforcement in ad hoc networks,” Computing Research Repository (CoRR), 2003. [Online]. Available: <http://arxiv.org/abs/cs/0307012>
- [22] S. Buchegger and J.-Y. L. Boudec, “Performance analysis of the CONFIDANT protocol: Cooperation of nodes fairness in dynamic ad-hoc networks,” in *Proceedings of the 3rd ACM International Symposium on Mobile Ad Hoc Networking and Computing (MobiHoc)*. ACM Press, 2002, pp. 226–236.
- [23] Q. He, D. Wu, and P. Khosla, “SORI: A secure and objective reputation-based incentive scheme for ad-hoc networks,” in *Proceedings of the Wireless Communications and Networking Conference (WCNC)*. IEEE Computer Society, 2002, pp. 825–830.
- [24] M. Ettus, “System capacity, latency, and power consumption in multihop-routed SS-CDMA wireless networks,” in *Proceedings of the Radio and Wireless Conference (RAWCON)*. IEEE Computer Society, 1998, pp. 55–58.
- [25] S. Singh, M. Woo, and C. S. Raghavendra, “Power-aware routing in mobile ad hoc networks,” in *Proceedings of the 4th annual ACM/IEEE international conference on Mobile computing and networking (MOBI-COM)*. ACM Press, 1998, pp. 181–190.
- [26] V. Rodoplu and T. H. Meng, “Minimum energy wireless networks,” *IEEE Journal on Selected Areas in Communications*, vol. 17, no. 8, pp. 1333–1344, Aug 1999.
- [27] J.-H. Chang and L. Tassiulas, “Energy conserving routing in wireless ad hoc networks,” in *Proceedings of the 19th Annual Joint Conference of the IEEE Computer and Communications Societies (INFOCOM)*. IEEE Computer Society, 2000, pp. 22–31.
- [28] R. C. Shah and J. M. Rabaey, “Energy aware routing for low energy ad hoc sensor networks,” in *Proceedings of the Wireless Communications and Networking Conference (WCNC)*. IEEE Computer Society, 2002, pp. 350–355.
- [29] J.-H. Chang and L. Tassiulas, “Maximum lifetime routing in wireless sensor networks,” *IEEE/ACM Transactions on Networking*, vol. 12, no. 4, pp. 609–619, Aug 2004.
- [30] F. Kargl, A. Klenk, S. Schlott, and M. Weber, “Advanced detection of selfish or malicious nodes in ad hoc networks,” in *Proceedings of 1st European Workshop on Security in Ad-Hoc and Sensor Networks (ESAS 2004)*, ser. Lecture Notes in Computer Science, C. Castelluccia, H. Hartenstein, C. Paar, and D. Westhoff, Eds., vol. 3313. Springer-Verlag, 2004, pp. 152–165.
- [31] S. Thrun, “Monte Carlo POMDPs,” in *Advances in Neural Information Processing Systems*, S. A. Solla, T. K. Leen, and K.-R. Müller, Eds., vol. 12. MIT Press, 1999, pp. 1064–1070.
- [32] E. Groes, H. J. Jacobsen, and B. Sloth, “Adaptive learning in extensive form games and sequential equilibrium,” *Economic Theory*, vol. 13, pp. 125–142, 1999.
- [33] R. S. Sutton and A. G. Barto, *Reinforcement Learning: An Introduction*. MIT Press, 1998.
- [34] R. S. Sutton, D. McAllester, S. Singh, and Y. Mansour, “Policy gradient methods for reinforcement learning with function approximation,” in *Advances in Neural Information Processing Systems*, S. A. Solla, T. K. Leen, and K.-R. Müller, Eds., vol. 12. MIT Press, 2000, pp. 1057–1063.
- [35] S. Haykin, *Neural Networks: A Comprehensive Foundation*. Prentice Hall, 1998.
- [36] J. Baxter and P. L. Bartlett, “Infinite-horizon policy-gradient estimation,” *Journal of Artificial Intelligence Research*, vol. 15, no. 4, pp. 319–350, 2001.
- [37] M. Bowling and M. Veloso, “Rational and convergent learning in stochastic games,” in *Proceedings of the 17th International Joint Conference on Artificial Intelligence (IJCAI)*. Morgan Kaufmann, 2001, pp. 1021–1026.
- [38] —, “Multiagent learning using variable learning rate,” *Artificial Intelligence*, vol. 136, no. 2, pp. 215–250, 2002.
- [39] D. Fudenberg and D. M. Kreps, “Learning in extensive-form games II, experimentation and Nash equilibrium,” Harvard University and Stanford University, Tech. Rep., 1994.
- [40] P. Nurmi, “A Bayesian framework for online reputation systems,” in *Proceedings of the Advanced International Conference on Telecommunications and International Conference on Internet and Web Applications and Services (AICT/ICIW)*. IEEE Computer Society, 2006.