Cost-effective Traffic Assignment for Multipath Routing in Selfish Networks

Fan Wu[1], **Sheng Zhong**[1], **and Jiqiang Liu**[1,2]

[1] Department of Computer Science and Engineering, State University of New York at Buffalo, USA
[2] School of Computer and Information Technology, Beijing Jiao Tong University, China
e-mail: {fwu2, szhong, jl278}@cse.buffalo.edu

Abstract Multipath routing has long been studied as an important routing strategy in networks. Many multipath routing protocols schedule traffic among multiple paths in order to distribute load. However, existing multipath routing protocols with traffic assignment require that all nodes in the network follow the protocol, which may not always be a valid assumption. In this paper, we propose a traffic assignment scheme to deal with selfish behavior, which is shown to be strategy-proof. Evaluations demonstrate that our scheme is very efficient.

Key words Routing, Traffic Assignment, Game Theory.

1 Introduction

Multipath Routing has long been studied as an important routing strategy in networks. It provides multiple paths for sending data from a source to a destination to exploit the resources of the underlying physical network. Previous research has demonstrated that multipath routing can achieve route resilience, higher aggregate bandwidth, smaller end-to-end delays, and better load balancing [1,2].

Multipath routing has been explored in both wired and wireless networks. In wired network, multipath routing is implemented as a feature of Asynchronous Transfer Mode (ATM) networks [3] and Open Shortest Path First (OSPF) protocol [4]. For Mobile Ad-Hoc Network (MANET), multipath routing is also extensively studied in recent years. A number of multipath routing protocols for MANETs have been proposed. Some of them [5–8] maintain multiple routes and utilize them only when the primary root fails. Others [9–11] further schedule traffic among multiple paths in order to distribute load. In this paper, we are mainly concerned with the latter, i.e., multipath routing protocols that assign the traffic among the multiple paths.

We note that the existing multipath routing protocols with traffic assignment require that all nodes in the network follow the prescribed protocol and cooperate with each other. However, this assumption is not valid when the network consists of selfish nodes [12–25]. Since forwarding flows depletes scarce resources such as power, and reduces available bandwidth to the node itself, when nodes in the network belong to different owners, they may not have incentive to forward others' flows. In this paper, we consider the selfish behavior of nodes in such networks. Specifically, a selfish node is an economically rational node whose objective is to maximize its own utility. So our question is how to design a multipath routing protocol such that selfish nodes will behave cooperatively.

To the best of our knowledge, there has not been any work addressing selfish behavior for multipath routing. However, there has been extensive study on traditional unicast and multicast in selfish networks. Considering the complexity and the subtlety of the incentive issues, many researchers apply game-theoretic techniques to analyze and design protocols in wireless and wired networks. In wireless network, various incentive-based approaches have been proposed to solve packet routing or forwarding problem [12–19]. Wang et al. [20] and Yuen et al. [21] investigated the problem of bandwidth allocation and multicast tree formation in overlay networks. Feigenbaum et al. [22,23] considered both unicast and multicast in Internet. Felegyhazi et al. [24] and Halldorsson et al. [25] studied the problem of sharing spectrum using game theory.

Although the methods mentioned above can not be directly used in the multipath routing scenario, we believe that we can develop a game-theoretic solution for multipath routing that can deal with the selfish behavior of nodes. To design a multipath routing protocol for selfish networks, instead of starting from scratch, we consider some existing multipath routing protocol and make it compatible with selfish behavior by redesigning its traffic assignment scheme. That is, we study how to assign the data traffic to the multiple paths established by a given multipath routing protocol between the source and the destination,

such that the participating selfish nodes will behave cooperatively. First, we give a game-theoretic model for this problem, which we call traffic assignment game. Then, we propose an efficient scheme for traffic assignment, which is shown to be *strategy-proof* in the above model. Here intuitively, the scheme being strategy-proof means that behaving cooperatively is to the best interest of every node, regardless of other nodes' behavior. Furthermore, our scheme is guaranteed to compute the lowest cost traffic assignment. Evaluations demonstrate that our scheme has very good performance.

The rest of this paper is organized as follows: In Section 2 we introduce some preliminaries. In Section 3, we present our traffic assignment game model. In Section 4, we go to the details of our traffic assignment scheme and prove its optimality and strategy-proofness. In Section 5, we show the results of evaluations. Finally, we conclude the paper in Section 6.

2 Technical Preliminaries

Before introducing our model, we need to recall some notations from mechanism design. In the classic model of mechanism design, there is a set of players $N = \{1, 2, \ldots, n\}$. Each player i has some private information t_i called type, which determines its preferences over different outcomes of a game. The players' type vector is denoted by $t = (t_1, t_2, \ldots, t_n)$. For each player $i \in N$, there is a set of available actions A_i. As a notational convention, a_{-i} represents the actions of all players except player i. Note that (a_i, a_{-i}) is an action profile, in which player i takes action a_i and the other players take actions a_{-i}. The action profile a decides the outcome $o(a)$ and payment $p(a)$ of the game, where $p(a) = (p_1(a), p_2(a), \ldots, p_n(a))$ is the vector of payment to each player. A valuation function $v_i(t_i, o(a))$ assigns a monetary value for player i to each possible output $o(a)$. Node i's utility u_i is a function as follows:

$$u_i(a) = v_i(o(a)) + p_i(a). \tag{1}$$

Given above notations, now we can define a very strong solution concept called *dominant strategy* [26].

Definition 1 *A dominant strategy of a player is one that maximizes its utility regardless of what strategies other players choose. Specifically, a_i is player i's dominant strategy if, for any $a'_i \neq a_i$ and any a_{-i},*

$$u_i(a_i, a_{-i}) \geq u_i(a'_i, a_{-i}). \tag{2}$$

A *direct-revelation* mechanism is a mechanism in which the only actions available to players are to make claims about their preferences to the mechanism. That is, the strategy of player i is reporting type $\hat{t_i} = s_i(t_i)$, based on its actual preferences t_i. A direct-revelation mechanism is *incentive-compatible* (IC) if reporting truthful information is a dominant strategy for each player. Another important property of a mechanism is *individual-rationality* (IR) — each player can always achieve at least as much expected utility from participation as without participation. Finally, we say a direct-revelation mechanism is *strategyproof* if it satisfies both IC and IR properties.

3 A Model of traffic assignment Game

We give the detail of our traffic assignment game's model in this section. Consider a network represented by $G = (V, E)$, where $V = \{v_1, v_2, \ldots, v_n\}$ is the set of nodes and $E = \{e_1, e_2, \ldots, e_m\} \subseteq V \times V$ is the set of undirected communication links, in which $e_k = v_i v_j$ means that node v_i and v_j can communicate with each other directly. Each node $v_i \in V$ has a fixed capacity C_i for sending and receiving data.

We model multipath routing with traffic assignment as a mechanism design problem, which we call the *traffic assignment game*. For a traffic assignment game, suppose that source node is S and destination node is D. Then the player set of the game is $V - \{S, D\}$. Each node $v_i \in V$ knows its cost function $f_i(x)$ and current available bandwidth $0 \leq b_i \leq C_i$, which are defined as its *type*. Formally, we have type $t_i =< f_i(x), b_i >$. The cost function $f_i(x)$ indicates the cost of forwarding one unit of traffic when x units of bandwidth has been used. It is a increasing and convex function, which grows slowly at the beginning, and then increases more and more rapidly. Intuitively, this means that the more bandwidth a node allocates for forwarding traffic, the less bandwidth it can use to send its own traffic. The cost includes expense of power consumed, losing in sacrificing bandwidth to send own traffic, and so on. So a node will get increasingly reluctant to sell its bandwidth when more and more bandwidth is used. Suppose a new flow request wants to go through node v_i with bandwidth requirement $q \leq b_i$. Then the overall cost of node v_i for forwarding this flow is

$$c_i = \int_{C_i - b_i}^{C_i - b_i + q} f_i(x)dx.$$

In this game, each player node v_i chooses an action a_i. The profile of all players' actions is denoted by $a = (a_i)_{v_i \in V - \{S,D\}}$. This action profile determines both the valuation function $v_i(o(a)) = -c_i(o(a))$ and the payment $p_i(a)$. So the utility of each node is:

$$u_i(a) = p_i(a) - c_i(o(a)). \tag{3}$$

4 Traffic Assignment Scheme

In this section, we propose our traffic assignment scheme and prove its strategy-proofness. Our scheme is designed for assigning traffic among multiple node-disjoint paths, which do not have any nodes in common except the source and destination. So our scheme can be used for any multipath routing protocol that schedules multiple node-disjoint paths (*e.g.*, [4,7,8]). In this paper, we assume that the network topology is biconnected — there exist at least two node-disjoint paths from any source node to any destination node.

4.1 Scheme

Given a new traffic demand q from a source node S to a destination node D, there is a set of node-disjoint paths $P = \{P_1, P_2, \ldots, P_m\}$ found by the multipath routing protocol (*e.g.*, [4,7,8]). Each player node v_i on any of these paths declares its cost function $f_i(x)$ and current available bandwidth b_i.

Algorithm 1 Traffic Assignment Algorithm

Input: Set of paths P, cost functions and available bandwidth $< f_i(x), b_i >_{v_i \in P_j \in P}$, and bandwidth requirement q.
Output: Traffic assignment $R = (r_1, r_2, \ldots, r_m)$.

$R = 0^m$
$W = \varnothing$
$\forall P_j \in P$, define $F_j(x) = \sum_{v_i \in P_j} f_i(C_i - b_i + x)$
while $q > \sum_{i=0}^{m} r_i$ **do**
  if $(P \neq \varnothing)$ **then**
    $k = \underset{P_j \in P}{argmin} F_j(0)$
    $c_m = F_k(0)$
  else
    $c_m = max$
  end if
  Compute the largest $r_1, r_2, \ldots, r_m$, s. t.
  $\exists c \leq c_m, \forall P_j \in W, F_j(r_j) = c$
  $\wedge \forall P_j \in W, \forall v_i \in P_j, r_j \leq b_i$
  $\wedge \sum_{P_j \in W} r_j \leq q$. (See Section 4.2 for detail.)
  $c_p = c$
  if $q > \sum_{P_j \in W} r_j$ **then**
    Move P_k from P to W
  end if
end while

After collecting all the information, the source node (or the destination node) computes the traffic assignment using Algorithm 1. Algorithm 1 first combines the cost functions declared by nodes on each path $P_i \in P$ to get a integrated path cost function $F_i(x)$. Then it turns to work on these integrated path cost functions. In each iteration, if there exist paths in the set P, it sets c_m to the smallest $F_j(0)$ among these paths; if there is no such path, then c_m is set as max (an upper bound of $F_j(x)$). Next, the algorithm tries to assign to each path in the set W as much traffic as possible, under the constraint that each path P_i has the same cost after allocating additional units of traffic and this cost is less than or equal to c_m. The result is a traffic assignment vector R. We will give how to calculate the vector R in next sub-section. If there is still some traffic has not been allocated, move P_k to the set W and repeat the above iteration. When the iteration stops, the vector R is the final assignment of traffic.

4.2 Calculating Vector R

To calculate the assigned traffic vector R in each iteration, we distinguish two cases:

1. $\sum_{P_i \in W} F_i^{-1}(c_m) \leq q$: Set $r_i = F_i^{-1}(c_m)$ for each path $P_i \in W$.
2. $\sum_{P_i \in W} F_i^{-1}(c_m) > q$: This time, we need to find a c, such that $c_p < c < c_m$, and $\sum_{P_i \in W}(F_i^{-1}(c)) = q$. Algorithm 2 is designed to calculate such value c. Then here $r_i = F_i^{-1}(c)$ for each path $P_i \in W$.

Algorithm 2 Calculating c

Input: $(F_1, F_2, \ldots, F_m)$, c_p, c_m, and q.
Output: c.

$top = c_m$
$bottom = c_p$
Set ϵ // Set precision factor.
while $bottom \leq top$ **do**
  $c = (top + bottom)/2$
  if $\sum_{P_i \in W}(F_i^{-1}(c)) > q + \epsilon$ **then**
    $top = c$
  else if $\sum_{P_i \in W}(F_i^{-1}(c)) < q - \epsilon$ **then**
    $bottom = c$
  else
    return c
  end if
end while
return not found

When using Algorithm 2, we can set ϵ to different values to satisfy various requirements of applications. The smaller ϵ is, the higher precision we have, but the amount of time consumed is also greater.

4.3 Payment to Each Node

To calculate the payment to each node v_i in each path in P, we call the Algorithm 1 twice. Suppose v_i is on path $P_j \in P$. The first execution of Algorithm 1 is exactly what we have described in Section 4.1. In the second execution, we remove the path P_j from the

network. Let R and R' be the traffic assignment computed by the two executions of Algorithm 1. Then the payment p_i to v_i is defined as follows:

$$\begin{aligned} p_i = & \sum_{P_k \in P - \{P_j\}} \int_{r_k}^{r'_k} F_k(x)dx \\ & - \sum_{v_h \in P_j - \{v_i\}} \int_0^{r_k} f_i(C_i - b_i + x)dx. \end{aligned} \tag{4}$$

If a node is not in any path in P, it does not receive any payment.

4.4 Optimality

Theorem 1 *Our traffic assignment scheme computes the most cost efficient traffic assignment if truthfulness is guaranteed.*

Proof Suppose that R is the traffic assignment computed by our algorithm and R' is any traffic assignment for traffic demand q. Then we have:

$$\sum_{P_j \in P} r'_j = \sum_{P_j \in P} r_j = q$$

We divide P into two subsets $P^{(1)}$ and $P^{(2)}$, such that $P = P^{(1)} \cup P^{(2)}$, $P^{(1)} \cap P^{(2)} = \emptyset$, and

$$\begin{cases} \forall P_j \in P^{(1)}, r_j \geq r'_j, \\ \forall P_j \in P^{(2)}, r_j < r'_j. \end{cases} \tag{5}$$

Now we consider the cost difference between the two traffic assignment.

$$\begin{aligned} & \sum_{P_j \in P} \left(\int_0^{r_j} F_j(x)dx - \int_0^{r'_j} F_j(x)dx \right) \\ = & \sum_{P_j \in P_1} \int_{r'_j}^{r_j} F_j(x)dx - \sum_{P_j \in P_2} \int_{r_j}^{r'_j} F_j(x)dx \\ \leq & \sum_{P_j \in P_1} c(r_j - r'_j) - \sum_{P_j \in P_2} c(r'_j - r_j) \\ = & c\left(\sum_{P_j \in P} r_j - \sum_{P_j \in P} r'_j \right) \\ = & 0 \end{aligned}$$

$$\Rightarrow \sum_{P_j \in P} \int_0^{r_j} F_j(x)dx \leq \sum_{P_j \in P} \int_0^{r'_j} F_j(x)dx$$

So the cost of traffic assignment R is less or equal to that of R'. ■

Now we consider the utility u_i of each node v_i in each path in P:

$$\begin{aligned} u_i &= p_i - c_i \\ &= \sum_{P_k \in P - \{P_j\}} \int_0^{r'_k} F_k(x)dx - \sum_{P_k \in P} \int_0^{r_k} F_k(x)dx. \end{aligned}$$

Since the traffic assignment computed by our algorithm is optimized, we have $u_i \geq 0$.

4.5 Strategy-Proofness

In our scheme, the actions available to each node in the network are to declare its private type. Obviously it is a direct-revelation mechanism. To show it has the incentive compatible (IC) property, we will prove that if our scheme is used, telling the truth is a dominant strategy.

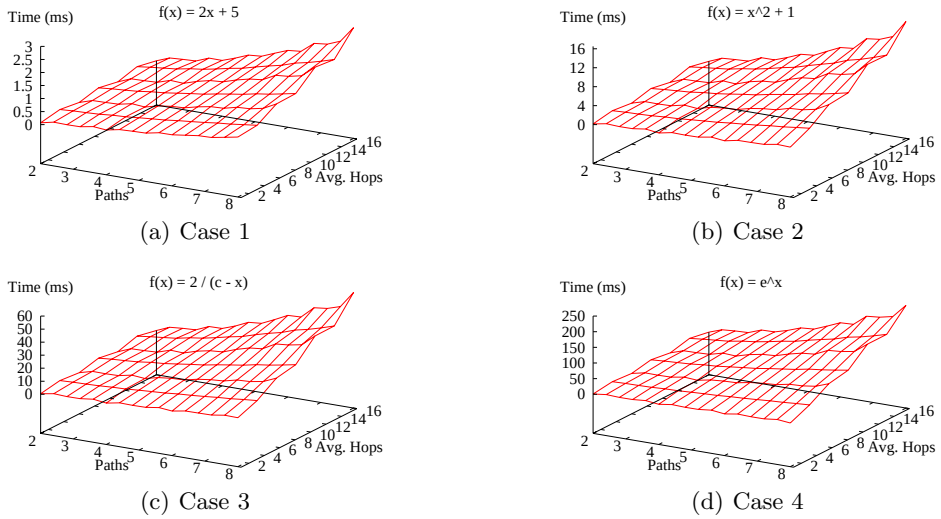
Theorem 2 *If our scheme is used, declaring the true type (cost function and available bandwidth) is a dominant strategy for each node.*

Proof We will show that a node v_i can not increase its utility by cheating. That is to say, truth telling is a dominant strategy. If the node v_i is not on any path in P, it will definitely get zero utility. If the node v_i is on one of the path P_j in P, we distinguish three cases:

1. The node v_i cheats to increase the amount of traffic passing through itself by $\Delta r_j > 0$. The traffic on path $P_k \in P - \{P_j\}$ is decreased by Δr_k (where some Δr_k may be less than or equal to 0). But the node's new utility $u'_i = p'_i - c'_i$ can not be more than u_i because:

$$\begin{aligned} u'_i - u_i &= (p'_i - c'_i) - (p_i - c_i) \\ &= (p'_i - p_i) - (c'_i - c_i) \\ &= \Big(\sum_{P_k \in P - \{P_j\}} \int_{r_k - \Delta r_k}^{r_k} F_k(x)dx \\ &\quad - \sum_{v_h \in P_j - \{v_i\}} \int_{r_j}^{r_j + \Delta r_j} f_h(C_h - b_h + x)dx \Big) \\ &\quad - \int_{r_j}^{r_j + \Delta r_j} f_i(C_i - b_i + x)dx \\ &= \sum_{P_k \in P - \{P_j\}} \int_{r_k - \Delta r_k}^{r_k} F_k(x)dx \\ &\quad - \int_{r_j}^{r_j + \Delta r_j} F_j(x)dx \end{aligned}$$

   Since $\forall P_k \in P - \{P_j\}, F_k(r_k) = F_j(r_j)$ when $r_k > 0$ and $\sum_{P_k \in P - \{P_j\}} \Delta r_k = \Delta r_j$, we have $u'_i - u_i \leq 0$.



(a) Case 1 (b) Case 2

(c) Case 3 (d) Case 4

Fig. 1 Computation time of using 4 kinds of cost functions as a function of number of paths and average hops.

2. The node v_i cheats to decrease the amount of traffic passing through itself by $\Delta r_j > 0$. The traffic on path $P_k \in P - \{P_j\}$ is increased by Δr_k. The node's new utility $u_i' = p_i' - c_i'$ can not be increased because:

$$\begin{aligned} u_i' - u_i &= (p_i' - c_i') - (p_i - c_i) \\ &= (p_i' - p_i) - (c_i' - c_i) \\ &= (-\sum_{P_k \in P - \{P_j\}} \int_{r_k}^{r_k + \Delta r_k} F_k(x)dx \\ &\quad + \sum_{v_h \in P_j - \{v_i\}} \int_{r_j - \Delta r_j}^{r_j} f_h(C_h - b_h + x)dx) \\ &\quad + \int_{r_j - \Delta r_j}^{r_j} f_i(C_i - b_i + x))dx \\ &= \int_{r_j - \Delta r_j}^{r_j} F_j(x)dx \\ &\quad - \sum_{P_k \in P - \{P_j\}} \int_{r_k}^{r_k + \Delta r_k} F_k(x)dx \end{aligned}$$

   Since $\forall P_k \in P - \{P_j\}, F_k(r_k) = F_j(r_j)$ when $r_k > 0$ and $\sum_{P_k \in P - \{P_j\}} \Delta r_k = \Delta r_j$, we have $u_i' - u_i \leq 0$.
3. The node v_i cheats, but does not change the amount of traffic passing through itself. Since both the payment to v_i and the cost for forwarding the traffic does not change, the node v_i still gets the same utility as that of telling truth. ■

From the definition of payment, we can see that whenever participating in the game, a node will get non-negative utility under our scheme. If a node stays out of the game, its utility will remain to be 0. So participating is not worse than staying out, which satisfies the individual rationality (IR).

Since our scheme satisfies both IC and IR, we have the following theorem:

Theorem 3 *The traffic assignment scheme presented in this paper is a strategy-proof mechanism.*

5 Evaluations

Now we evaluate the efficiency of our scheme in terms of communication overhead and computational overhead.

The overall communication overhead is $N_p N_h (L_f + L_b)$ bits, where N_p is the number of node-disjoint paths from S to D, N_h is the average number of hops of these paths, and L_f and L_b are the numbers of bits needed to encode the cost function and available bandwidth, respectively.

To evaluate the computational overhead, we have implemented our traffic assignment scheme and run it on a laptop with 1.4GHz Centrino CPU and 768MB memory. In our evaluations, we have tried various cost functions, including linear function (Case 1, $f(x) = 2x + 5$), quadratic function (Case 2, $f(x) = x^2 + 1$), reciprocal function (Case 3, $f(x) = \frac{2}{C-x}$), and exponential function (Case 4, $f(x) = e^x$). We test the performance of our scheme in a series of setups: the number of candidate node-disjoint paths ranges from 2 to 8, and the average hops per path ranges from 1 to 16. We set ϵ to 10^{-5}. This precision is much more than needed in most applications.

Figure 1 shows the performance for each cost function under different number of paths and different average number of hops. Generally, running time of

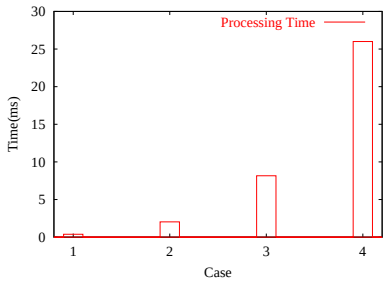


Fig. 2 Running time of using 4 kinds of cost function with 3 paths and 16 average hops.

Case 1 and Case 2 is less than that of Case 3 and Case 4. This is because it is easier to compute the inverse function of $F_j(x) = \sum_{v_i \in P_j} f_i(x)$ when $f_i(x)$ is a linear or quadratic function. For the other two cases, binary search is needed to compute $F_j^{-1}(x)$, which needs more time.

Figure 2 compares running time of our scheme using four different cost functions illustrated above, when the number of candidate paths is 3 and the average hop number is 16. Even in Case 4 we can finish the computation in 26ms. So the computational overhead is low when our scheme is used.

From these results of evaluation, we can see that our scheme is very efficient.

6 Conclusion

In this paper, we propose a game-theoretic solution for multipath routing to deal with the selfish behavior of nodes. Our scheme can be used to modify any existing multipath routing protocol that schedules traffic among node-disjoint paths such that the protocol becomes incentive compatible. The evaluation results show that our scheme is very efficient.

References

1. Suzuki, H., Tobagi, F.A.: Fat bandwidth reservation scheme with multi-link and multi-path routing in atm networks. In: IEEE INFOCOM. (1992)
2. Cidon, I., Rom, R., Shavitt, Y.: Analysis of multi-path routing. IEEE/ACM Tran. on Networking **7**(6) (1999) 885–896
3. CORPORATE The ATM Forum: (ATM user-network interface specification (version 3.0))
4. Moy, J.: OSPF (version 2), RFC 2328 (1998)
5. Park, V.D., Corson, M.S.: A highly adaptive distributed routing algorithm for mobile wireless networks. In: IEEE INFOCOM. (1997)
6. Lee, S., Gerla, M.: AODV-BR: Backup routing in ad hoc networks. In: IEEE WCNC. (2000)
7. Marina, M., Das, S.: On-demand multi path distance vector routing in ad hoc networks. In: IEEE ICNP. (2001)
8. Ye, Z., Krishnamurthy, S.V., Tripathi, S.K.: A framework for reliable routing in mobile ad hoc networks. In: IEEE INFOCOM. (2003)
9. Lee, S., Gerla, M.: Split multipath routing with maximally disjoint paths in ad hoc networks. In: IEEE ICC. (2001)
10. Papadimitratos, P., Haas, Z., Sirer, E.: Path-set selection in mobile ad hoc networks. In: ACM MobiHoc. (2002)
11. Pearlman, M.R., Haas, Z.J., Sholander, P., Tabrizi, S.S.: On the impact of alternate path routing for load balancing in mobile ad hoc networks. In: ACM MobiHoc. (2000)
12. Anderegg, L., Eidenbenz, S.: Ad hoc-vcg: a truthful and cost-eftcient routing protocol for mobile ad hoc networks with seltsh agents. In: ACM MobiCom. (2003)
13. Srinivasan, V., Nuggehalli, P., Chiasserini, C.F., Rao, R.: Cooperation in wireless ad hoc wireless networks. In: IEEE INFOCOM. (2003)
14. Wang, W., Li, X.Y., Wang, Y.: Truthful multicast in selfish wireless networks. In: ACM MobiCom. (2004)
15. Zhong, S., Li, L., Liu, Y.G., Yang, Y.R.: On designing incentive-compatible routing and forwarding protocols in wireless ad-hoc networks. In: ACM MobiCom. (2005)
16. Zhong, S., Chen, J., Yang, Y.R.: Sprite, a simple, cheat-proof, credit-based system for mobile ad-hoc networks. In: IEEE INFOCOM. (2003)
17. Wang, W., Eidenbenz, S., Wang, Y., Li, X.Y.: Ours: Optimal unicast routing systems in non-cooperative wireless networks. In: ACM MobiCom. (2006)
18. Salem, N., Buttyan, L., Hubaux, J., Jakobsson, M.: A charging and rewarding scheme for packet forwarding in multi-hop cellular networks. In: ACM MobiHOC. (2003)
19. Eidenbenz, S., Resta, G., Santi, P.: Commit: A sender-centric truthful and energy-efficient routing protocol for ad hoc networks with selfish nodes. In: IEEE IPDPS. (2005)
20. Wang, W., Li, B.: Market-driven bandwidth allocation in selfish overlay networks. In: IEEE INFOCOM. (2005)
21. Yuen, S., Li, B.: Strategyproof mechanisms for dynamic multicast tree formation in overlay networks. In: IEEE INFOCOM. (2005)
22. Feigenbaum, J., Papadimitriou, C.H., Shenker, S.: Sharing the cost of multicast transmissions. J. Computer and System Sciences **63**(1) (2001) 21–41
23. Feigenbaum, J., Papadimitriou, C., Sami, R., Shenker, S.: A bgp-based mechanism for lowest-cost routing. In: ACM PODC. (2002)
24. Felegyhazi, M., Hubaux, J.P.: Wireless Operators in a Shared Spectrum. In: IEEE INFOCOM. (2006)
25. Halldósson, M.M., Halpern, J.Y., Li, L.E., Mirrokni, V.S.: On spectrum sharing games. In: ACM PODC. (2004)
26. Osborne, M.J., Rubenstein, A.: A Course in Game Theory. The MIT Press (1994)