Finding Cheap Routes in Profit-Driven Opportunistic Spectrum Access Networks: A Truthful Mechanism Design Approach

Tao Shu and Marwan Krunz

***Abstract*—In this paper, we explore the economic aspects of routing/relaying in a *profit-driven* opportunistic spectrum access (OSA) network. In this network, primary users lease their licensed spectrum to secondary radio (SR) providers, who in turn provide opportunistic routing/relaying service to end users if this service is profitable, i.e., if the payment offered by the end user (a.k.a the *price*) exceeds the SR's relaying spectrum *cost*. This cost is considered private information known only to SRs. Therefore, the end user has to rely on costs reported by SRs to determine his routing and payment strategy. The challenge comes from the selfish nature of SRs; an SR may exaggerate his cost to achieve greater profit. To give incentive to an SR to report the true cost, the payment must typically be higher than the actual cost. However, from the end user's perspective, "overpayment" should be avoided as much as possible. So we are interested in the "optimal" route selection and payment determination mechanism that minimizes the price of the selected route while simultaneously guaranteeing truthful cost reporting by SRs. We formulate this problem as finding the least-priced path (LPP), and we investigate it without and with link capacity constraints. In the former case, polynomial-time algorithm is developed to find LPP and calculate its truthful price. In the latter case, we show that calculating the truthful price of the LPP is in general computationally infeasible. Consequently, we consider a sub-optimal but computationally feasible approximate solution, which we refer to as truthful low-priced path (LOPP) routing. A polynomial-time algorithm is proposed to find the LOPP and efficiently calculate its truthful price. A payment materialization algorithm is also developed to guarantee truthful capacity reporting by SRs. The effectiveness of our algorithms in terms of price saving is verified through extensive simulations.**

***Index Terms*—Truthful mechanism design, opportunistic spectrum access, least-priced-path routing.**

I. Introduction

A. Motivation

The benefit of *opportunistic spectrum access* (OSA) as a means of improving spectrum utilization is now well recognized [20]. OSA aims at opening under-utilized portions of the licensed spectrum for secondary re-use, provided that the transmissions of secondary radios (SRs) do not cause harmful interference to the communications of primary users (PUs).

Naturally, profit is a critical driving force behind the realization of OSA. Under the assumption of economically rational users, a PU has interest in opening his idle spectrum for secondary re-use only if he can make a profit from such an action. To this end, an SR is typically charged for leasing the spectrum on a temporary basis. Such economic consideration has been reflected in recent OSA studies. For example, the spectrum auctions in [23] and [8] considered the situation where SRs buy spectrum through a bidding process. The spectrum-leasing architecture in [14] requires SRs to subscribe to (and pay for) the spectrum-status information broadcasted by a spectrum server. The IEEE 802.22 WRAN standard is based on an infrastructure-type architecture, which by design is suitable for implementing fee-based OSA services. In addition to providing incentive to PUs, profit is also an important factor for SR service providers, who would be willing to transport the traffic of other users if they can make a profit, i.e., if the payment received by the servicing node (a.k.a, the *price* of the service) exceeds the spectrum cost paid by the SR.

In contrast to previous works that studied the cost aspects of acquiring spectrum, in this paper we study its "end system" perspective and consider the implication of the *for-profit* nature of SRs. We focus on the economic aspects of the routing problem in a profit-driven OSA network. Specifically, we consider the situation where an end user wishes to purchase a route to a destination. Intermediate SRs along this route are willing to relay the traffic from this source for a fee. The problem for the source is to decide the cheapest route, i.e., the one that minimizes the source's total payment to relaying nodes (equivalently, the price of the route). We refer to this problem as the *least-priced path (LPP)* problem. Note that even though we assume the source pays for the route, the problem does not lose its generality if the destination is the one who makes the payment.

At a first glance, the LPP problem may seem easy to solve using the following naive method. The source would ask intermediate SRs to report their costs. The source would then choose the best path to the destination with respect to the reported cost. Each SR along the selected path would be paid the equivalent of its reported cost. The problem with this method is that intermediate SRs may exaggerate their claimed costs for the purpose of getting higher profits. This is especially true when intermediate SRs belong to a different administrative domain than the source, and thus are inclined to act selfishly to maximize their profit. As a result, the source could end up paying an unnecessarily high price for the route.

The above selfish behavior has been addressed in the literature under a different setup, namely, finding the least-cost path (LCP) (e.g., see [13], [7], [6], [17]). The basic idea in the LCP algorithm is to design a "truthful" payment mechanism, which guarantees that mis-claiming the cost will not increase

A preliminary version of this work was presented at the *IEEE INFOCOM 2010 Conference*, San Diego, CA, March 15-19, 2010.

the payment made to the relaying node. As a result, such a node has no incentive to exaggerate its true cost. The LCP can be subsequently constructed based on the reported costs.

In contrast to the LCP problem, the LPP problem studied in this paper takes into account the following three new aspects that arise in the OSA context. First, instead of minimizing the *cost* of the route (as reported by relay nodes), we aim at minimizing its *price* (payment made by end user). This is more attractive to an end user, because price is the actual expense the end user has to pay. As will become clear shortly, under the truthfulness requirement, there is no straightforward conversion between these two objectives; an LCP may not have the least price. Hence, a new formulation is required for the LPP problem.

Second, in contrast to the LCP formulation, where the cost of an SR node is constant, in our setup cost is modeled as a random variable. The node cost in OSA represents the monetary value the SR node pays to the PU. Such a value changes with spectrum supply/demand dynamics in the vicinity of a node. When supply is tight, an SR will have to pay more to access the spectrum. To cope with these variations, we consider a randomized routing strategy, which adapts to the dynamics of the node cost, with the goal of minimizing the expected price of the route.

Third, when constructing a route, we consider the transport capacity limit of each node. By being restricted to a secondary role, an SR cannot guarantee it can always acquire the required amount of spectrum. Therefore, an end-to-end flow may have to be split into multiple sub-flows, leading to multi-path routing. This is in contrast to the single-path situation considered in the LCP problem, where no capacity constraint is imposed on relaying nodes, so a flow can always be accommodated by a single path.

B. Contributions and Paper Organization

In this paper, we model truthful LPP routing as a game theoretic *mechanism design* problem [13]. We provide a unified formulation and solutions that address the aforementioned three considerations. Our investigation is divided into two parts. In the first part, we obtain the LPP and its truthful price without imposing a node-capacity constraint. This simplified formulation applies to the scenario where the rate demand is relatively low, such that relaying SRs can always support it. In the second part, we consider the problem under a given source rate demand and given capacity constraints at intermediate nodes. We derive the analytical solution for the LPP and its truthful payment. We show that the LPP in this case is a multi-path route, and in general calculating the truthful payments is computationally infeasible. Consequently, we consider a sub-optimal but computationally feasible version of the problem, namely, truthful low-priced path (LOPP) routing. The price of the LOPP is not necessarily the lowest, but is still significantly lower than the price of the path found by other truthful routing algorithms such as LCP. More importantly, the truthful payment under the LOPP algorithm can be efficiently computed. Our LOPP construction is the first truthful multi-path routing algorithm in the literature that accounts for general non-node-disjoint routes. When the routes under consideration are node-disjoint, the LOPP algorithm returns the truthful LPP. Under a node-capacity constraint, capacity information must be collected. So we develop a payment enforcement algorithm that guarantees truthful capacity reporting by relay nodes. The complete LOPP routing mechanism guarantees that misreporting the node's cost and capacity does not lead to a higher profit for that node.

Our work extends Myerson's optimal auction problem [11], in which bidders correspond to paths in our LPP formulation. In Myerson's problem, bidders are independent; no matter how a bidder changes his bid, he cannot change other bidders' bids. In contrast, in our LPP problem, paths need not be node-disjoint. As a result, when a node that belongs to multiple paths changes its claimed cost, the claimed costs of all involved paths will also change. In other words, the players in our problem are dependent. Therefore, the results of Myerson's work cannot be directly applied to our problem.

The remainder of this paper is organized as follows. Related work is reviewed in Section II. We formulate the LPP problem in Section III. The problem is investigated without and with node-capacity constraints in Sections IV and V, respectively. Simulation results are presented in Section VI. The paper is concluded in Section VII.

II. Related Work

In [5] the authors studied the minimization of the expected price of a *single* path under no node capacity constraint. Our results in Section IV are compatible with theirs. The main difference is that, by definition, their setup targets only single-path routes, whereas ours starts from a more general setting that allows for multi-path routing. This change in formulation is nontrivial, because now we need to explicitly account for the inter-dependence between paths. We formally prove that in the absence of capacity constraints, the LPP reduces to a single path. Our key contribution here is in proving that the outcome of the presented algorithm is not only optimal among the set of single paths, but also optimal among all possible combinations of paths. As to the second part of our work, i.e., LPP under node capacity constraints, to the best of our knowledge, this aspect has not been addressed before.

Aside from [5], other related works on truthful routing focused on the LCP problem rather than the LPP problem. The LCP problem was first introduced by Nisan and Ronen [13], who solved the truthful unicast LCP routing problem by applying the celebrated Vickrey-Clarke-Groves (VCG) mechanism. In [13], the cost of an agent (a node or an edge) is used as the agent's weight in the graph. The payment p^e to an agent e is 0 if e is not on the LCP and $p^e = d_{G|e=\infty} - d_{G|e=0}$ if e is on that path. Here, $d_{G|e=\infty}$ is the cost of the LCP on the graph that excludes e, and $d_{G|e=0}$ is the cost of the LCP when e is included and its cost is zero.

Several follow-up works extended the basic VCG algorithm [13] to various networking environments. This includes the ad hoc-VCG [1], the VCG-based BGP [6], the multicast version of VCG [18], and more recently VCG for opportunistic routing [19]. The work in [9] formulated the multi-path LCP problem but did not provide a solution. The work in [15] gave initial results for this problem by only considering the special case when all paths in the graph are disjoint. Some works, e.g., [7], focused on complexity issues in the VCG payment calculation. Other works, e.g., [22][16][17], went beyond the routing layer and encompassed a cross-layer methodology in studying the LCP problem.

The overpayment issue in VCG-based LCP routing was first noticed by Archer and Tardos [2]. They investigated the *frugal path* problem (FPP), which aims at designing a mechanism that selects a path and induces truthful cost revelation, but without paying high price. They also showed that no reasonable mechanism can always avoid paying a high premium to induce truthtelling. Subsequent works on FPP, e.g., [6] [4], provided bounds on the route price for general truthful routing mechanisms. Rather than following a bounding approach, the LPP problem addressed in this paper differs from FPP in that it explicitly minimizes the price of the route in a given OSA network.

When truthfulness is not a concern, various pricing mechanisms have been studied with the goal of optimizing the social welfare of a wireless ad hoc network. For example, the authors in [10] proposed a centralized two-fold pricing mechanism that accounts for the relay and interference between nodes in routing, with the purpose of maximizing the aggregate network utility. The authors in [12] proposed distributed pricing algorithms to achieve the same goal. they also studied the problem of balancing profits among nodes by optimizing a profit fairness metric. In all these works, it is assumed that a node truthfully reports information to the algorithm.

III. System Model and Problem Formulation

A. Preliminaries

The LPP problem is well suited for analysis by means of *mechanism design*, a branch of game theory. We start by briefly reviewing a few concepts from mechanism design. We then describe our model and formulate the problem using mechanism design's terminology.

A mechanism design problem considers a game of n *agents*, each with his own strategy set. Each agent i, $1 \leq i \leq n$, has some private information c_i (known only to agent i), called its true *type*. We consider the *direct revelation* strategy, in which agents simultaneously report their types to the mechanism. Such a strategy is simple to implement in practice and its communication overhead is small, making it scalable for large networks. Denote the reported type of agent i by $\tilde{c}_i$ ($\tilde{c}_i$ may be different from c_i) and the vector of reported types from all agents as $\tilde{\mathbf{c}} = (\tilde{c}_1, \ldots, \tilde{c}_n)$. The mechanism takes $\tilde{\mathbf{c}}$ as input and computes a response $\mathbf{X}(\tilde{\mathbf{c}}) = (x_1(\tilde{\mathbf{c}}), \ldots, x_n(\tilde{\mathbf{c}}))$ and a payment to agents $\mathbf{p}(\tilde{\mathbf{c}}) = (p_1(\tilde{\mathbf{c}}), \ldots, p_n(\tilde{\mathbf{c}}))$. The response $\mathbf{X}(\tilde{\mathbf{c}})$ is the action each agent is going to take, given the input $\tilde{\mathbf{c}}$. Given $\mathbf{X}$ and the type c_i, agent i's *cost function* is decided by a real-valued function $\vartheta_i(c_i, \mathbf{X})$. Accordingly, agent i's profit for reporting $\tilde{c}_i$ is given by $U_i \overset{\text{def}}{=} p_i(\tilde{\mathbf{c}}) - \vartheta_i(c_i, \mathbf{X})$. An agent is said to be rational if his reported $\tilde{c}_i$ maximizes his profit.

The main task of a mechanism design problem is to determine the functions $\mathbf{X}(\tilde{\mathbf{c}})$ and $\mathbf{p}(\tilde{\mathbf{c}})$ that maximize some social interest (e.g., social welfare) of the system. Usually, the following properties are desired in the mechanism:

1) Incentive compatible (IC): A mechanism is IC if each rational agent maximizes his profit when reporting his true type c_i.
2) Individually rational (IR): A mechanism is IR if each agent's profit for participating in the game is nonnegative.

When a mechanism is both IC and IR, we say it is *truthful*.

B. Network Model and Problem Formulation

We consider an OSA network whose topology is defined by a directed graph $G = (V, E)$, where V and E are the set of SR nodes and the set of directed links between SRs, respectively. SR nodes correspond to agents in the mechanism design framework. We assume there is no collusion between SRs, i.e., SRs react independently (no cooperation and information exchange between SRs) when bidding for PU spectrum and when providing service to end users. Such a scenario happens, for example, when SRs belong to different service providers. Each SR node j in V can access b_j amount of spectrum by paying a cost c_j to the PU for each transmitted packet. We do not make any assumption on the structure of b_j (e.g., deterministic or random). Our algorithm requires only the instantaneous value (sample) of b_j in the current session. The cost c_j results from, say, winning a bid in a spectrum auction or an agreed upon rent in a spectrum lease, and thus is considered private information (i.e., the true type), only known to node j. Because c_j depends on the spectrum supply/demand dynamics around node j, it is modeled as a random variable. We assume that $\{c_j : j \in V\}$ are independent. This assumption is supported by the location-dependent nature of spectrum supply/demand dynamics, and is reasonable when there is no collusion between SRs.

Now consider a traffic flow that originates from a source node s and terminates at a destination node d. The flow lasts for multiple sessions. Let $n \overset{\text{def}}{=} |V - \{s, d\}|$ ($|.|$ is the cardinality of a set), $m \overset{\text{def}}{=} |E|$, and label the nodes in $V - \{s, d\}$ as $j = 1, \ldots, n$. Let the set of all possible paths from s to d be $\mathbf{R}_{sd}$. Let $N \overset{\text{def}}{=} |\mathbf{R}_{sd}|$. Note that the enumeration of paths from s to d is intended only for the problem formulation. Our final algorithms do not require such enumeration. The routing mechanism operates on a session-by-session basis. At the beginning of a session, each node j in V reports a cost $\tilde{c}_j$ to the routing mechanism. A truthful mechanism should guarantee that $\tilde{c}_j = c_j$, i.e., the mechanism must satisfy the IC and IR constraints. We will formulate these constraints after we finish describing the general operation of the mechanism.

The mechanism takes as input the reported costs $\tilde{\mathbf{c}} = (\tilde{c}_1, \ldots, \tilde{c}_n)$ from the nodes in $V - \{s, d\}$, and computes an outcome consisting of a routing vector $\mathbf{X} = (x_1, \ldots, x_N)$ and a payment vector $\mathbf{p} = (p_1, \ldots, p_n)$. In general, we allow multi-path routing. So an element in $\mathbf{X}$, say x_i, $1 \leq i \leq N$, represents the fraction of traffic that will be carried over path i in $\mathbf{R}_{sd}$ during the current session, and p_j, $1 \leq j \leq n$, is the payment to node j for every packet delivered by paths in

$\mathbf{R}_{sd}$. Because x_i and p_j are outputs of the mechanism, we write them as functions of the reported costs, i.e., $x_i \overset{\text{def}}{=} x_i(\tilde{\mathbf{c}})$ and $p_j \overset{\text{def}}{=} p_j(\tilde{\mathbf{c}})$.

In practice, the mechanism can be executed either by the end user or by a centralized server. In the former case, the source node is responsible for collecting cost reports from individual SRs and computing the path vector and payments. Such an implementation is suitable for an end user of sufficient power/computation capability. A centralized server implementation is more appropriate for resource-constrained users. To simplify the presentation, in our subsequent analysis we use the notation in Table I.

We stick to the conventional economics approach of a Bayesian optimal mechanism design and assume that the probability density function (p.d.f.) of c_j, denoted by f_j, is known to the "mechanism", i.e., the entity that computes the routing and payment vectors and all the nodes in V. Let the domain of f_j be $D_j \overset{\text{def}}{=} [v_j, w_j]$, where $v_j \geq 0$, $w_j \geq 0$, and $v_j \leq w_j$. In practice, for a truthful mechanism, f_j and D_j can be constructed based on historical data of reported costs. In our analysis, we assume the mechanism has perfect knowledge of f_j, but we relax this assumption in our simulations. We further assume that c_j does not change during a session, but may change from one session to another. Here, a session represents continuous transmission of a burst of packets. For two nodes j and k, f_j and f_k are independent but not necessarily identical. This assumption captures the fact that in an OSA system, different nodes may experience heterogeneous spectrum availabilities, and thus their costs are stochastically non-identical. We assume that interference has been taken care of by the underlying spectrum allocation mechanism, so that interfering nodes operate over different frequency channels. This assumption is commonly supported in spectrum auction and leasing mechanisms, e.g., the algorithm in [23] takes as input the interference graph to avoid selling a channel to two nodes that interfere with each other. We also assume that even though a node is only allowed to transmit over channels it has acquired, it can tune to any channel for reception.

Even though several source-destination flows may exist in the network, these flows join the network sequentially. Accordingly, our mechanism considers source-destination pairs, one at a time. In this case, for a new incoming flow, b_j represents the *residual* spectrum at node j, i.e., the total idle spectrum available to SR j minus the spectrum that has been allocated to ongoing flows that are traversing through node j.

The truthful LPP routing problem is formulated as follows. The objective is to minimize the expected price that s needs to pay for each packet it sends in a session, i.e.,

$$\text{minimize}_{(\mathbf{X},\mathbf{p})} \quad \left\{ \Gamma_s(\mathbf{X},\mathbf{p}) \overset{\text{def}}{=} \int_{\mathbf{D}} \sum_{j \in V\backslash\{s,d\}} p_j(\tilde{\mathbf{c}})\mathbf{f}(\tilde{\mathbf{c}})d\tilde{\mathbf{c}} \right\} \tag{1}$$

subject to the following constraints:

Incentive Compatibility (IC) Constraint: This condition motivates a rational node j to report its true cost to the mechanism, i.e., it ensures that $\tilde{c}_j = c_j$. The basic idea is to design $\mathbf{X}$ and $\mathbf{p}$ in such a way that, if a number $z \in D_j$ is used as the unit spectrum cost to calculate node j's profit, i.e., $c_j = z$, then reporting $\tilde{c}_j = z'$, where $z' \in D_j$ and $z' \neq z$, does not lead to higher profit for node j than reporting $\tilde{c}_j = z$. This requirement is met for $\forall z$ and $z' \in D_j$ and $\forall j \in V\backslash\{s,d\}$. Mathematically, the IC constraint is formulated as follows

notation	definition
$\mathbf{c}_{-j}$	$(c_1,\ldots,c_{j-1},c_{j+1},\ldots,c_n)$
$(z,\mathbf{c}_{-j})$	$(c_1,\ldots,c_{j-1},z,c_{j+1},\ldots,c_n)$
$\tilde{\mathbf{c}}_{-j}$	$(\tilde{c}_1,\ldots,\tilde{c}_{j-1},\tilde{c}_{j+1},\ldots,\tilde{c}_n)$
$(z,\tilde{\mathbf{c}}_{-j})$	$(\tilde{c}_1,\ldots,\tilde{c}_{j-1},z,\tilde{c}_{j+1},\ldots,\tilde{c}_n)$
$\mathbf{D}_{-j}$	$\bigcup_{1<k<n,k\neq j} D_k$
$\mathbf{D}$	$\bigcup_{1\leq k\leq n} D_k$
$\mathbf{f}_{-j}(\tilde{\mathbf{c}}_{-j})$	$\prod_{1<k<n,k\neq j} f_k(\tilde{c}_k)$
$\mathbf{f}(\tilde{\mathbf{c}})$	$\prod_{1<k<n} f_k(\tilde{c}_k)$
$d\tilde{\mathbf{c}}_{-j}$	$d\tilde{c}_1\ldots d\tilde{c}_{j-1}d\tilde{c}_{j+1}\ldots d\tilde{c}_n$
$d\tilde{\mathbf{c}}$	$d\tilde{c}_1\ldots d\tilde{c}_n$

TABLE I
NOTATION USED IN THE ANALYSIS.

$$\begin{aligned} & U_j(\mathbf{X},\mathbf{p},z) \\ \overset{\text{def}}{=} & \int_{\mathbf{D}_{-j}} \left[p_j(z,\tilde{\mathbf{c}}_{-j}) - z \sum_{i \in \mathbf{R}_{sd}^{(j)}} x_i(z,\tilde{\mathbf{c}}_{-j}) \right] \mathbf{f}_{-j}(\tilde{\mathbf{c}}_{-j})d\tilde{\mathbf{c}}_{-j} \\ \geq & \int_{\mathbf{D}_{-j}} \left[p_j(z',\tilde{\mathbf{c}}_{-j}) - z \sum_{i \in \mathbf{R}_{sd}^{(j)}} x_i(z',\tilde{\mathbf{c}}_{-j}) \right] \mathbf{f}_{-j}(\tilde{\mathbf{c}}_{-j})d\tilde{\mathbf{c}}_{-j} \\ & \text{for } \forall z, \forall z' \in D_j, \forall j \in V\backslash\{s,d\} \end{aligned} \tag{2}$$

where $\mathbf{R}_{sd}^{(j)}$ is the subset of paths in $\mathbf{R}_{sd}$ that traverse node j. In (2), the L.H.S. of the inequality is the expected profit node j receives for each packet delivered over the route when it reports $\tilde{c}_j = z$, and the R.H.S. is the node's profit when it reports $\tilde{c}_j = z'$. Because a mechanism that satisfies the IC constraint (2) leads to $\tilde{c}_j = c_j$ for $\forall j \in V\backslash\{s,d\}$, the notations c_j and $\tilde{c}_j$ become inter-changeable. Our subsequent presentation will be based on $\tilde{c}_j$'s only, because they are the actual inputs to the mechanism.

Individual Rationality Constraint: This constraint requires that node j participates in packet relaying only if its expected profit is nonnegative, i.e.,

$$U_j(\mathbf{X},\mathbf{p},\tilde{c}_j) \geq 0 \quad \forall j \in V\backslash\{s,d\}. \tag{3}$$

Multi-path Constraint: This constraint says that the sum of the fractions of traffic carried over various paths must equal to the total traffic volume, i.e.,

$$\sum_{i=1}^{N} x_i(\tilde{\mathbf{c}}) = 1. \tag{4}$$

Node Capacity Constraint: The aggregate traffic of the sub-flows that traverse the same node should not exceed the node's capacity, i.e.,

$$\sum_{i \in \mathbf{R}_{sd}^{(j)}} x_i(\tilde{\mathbf{c}})R \leq b_j, \forall j \in V \tag{5}$$

where R is the flow rate demand.

Note that in the above formulation, the node cost is defined for each packet relayed by that node, whereas payment, price, and profit are with respect to each packet delivered over the (multi-path) route. Hereafter, we use the notation i and j to refer to a path and a node, respectively, unless indicated otherwise.

IV. LPP Without Node-Capacity Constraints

The main difficulty in solving the LPP problem is that the paths between s and d may not be node-disjoint. Therefore, the traffic volumes carried by different nodes are not independent. This dependence, which appears in the IC and the node-capacity constraints, prevents us from directly using Myerson's optimal auction theory [11]. In this section, we first consider a simplified version of the LPP formulation, in which we ignore the node-capacity constraint in (5). We refer to this problem as the *simplified LPP problem*. The resulting mechanism will still be truthful, because the IC and IR constraints are still being accounted for.

Our analysis of the simplified LPP problem proceeds as follows. Consider the IC constraint in (2). For node j, the fraction of traffic it expects to relay given its reported cost $\tilde{c}_j$ can be calculated as

$$Q_j(\tilde{c}_j) \overset{\text{def}}{=} \int_{\mathbf{D}_{-j}} \sum_{i\in\mathbf{R}_{sd}^{(j)}} x_i(\tilde{\mathbf{c}})\mathbf{f}_{-j}(\tilde{\mathbf{c}}_{-j})d\tilde{\mathbf{c}}_{-j}. \tag{6}$$

Based on $Q_j(\tilde{c}_j)$, we have the following lemma.

Lemma 1: The IC constraint in (2) is equivalent to satisfying both of the following two conditions:
(1) Monotonicity: For $\forall\tilde{c}_j^{(1)}$ and $\tilde{c}_j^{(2)} \in D_j$, if $\tilde{c}_j^{(1)} \geq \tilde{c}_j^{(2)}$, then $Q_j(\tilde{c}_j^{(1)}) \leq Q_j(\tilde{c}_j^{(2)})$;
(2) $U_j(\mathbf{X},\mathbf{p},\tilde{c}_j) = \int_{\tilde{c}_j}^{w_j} Q_j(\tau_j)d\tau_j + U_j(\mathbf{X},\mathbf{p},w_j)$, $\forall\tilde{c}_j \in D_j$.

The proof of Lemma 1 is provided in Appendix A. Based on Lemma 1, the LPP problem without node-capacity constraints can be re-formulated as follows.

Lemma 2: The optimal route vector $\mathbf{X}$ is the solution to the following optimization problem:

$$\begin{array}{ll} \text{minimize} & \sum_{1\leq j\leq n} \int_{\mathbf{D}} \left[\tilde{c}_j + \frac{F_j(\tilde{c}_j)}{f_j(\tilde{c}_j)}\right] \sum_{i\in\mathbf{R}_{sd}^{(j)}} x_i(\tilde{\mathbf{c}})\mathbf{f}(\tilde{\mathbf{c}})d\tilde{\mathbf{c}} \\ s.t. & \\ & \sum_{i=1}^{N} x_i(\tilde{\mathbf{c}}) = 1 \end{array} \tag{7}$$

where $F_j(\tilde{c}_j) = \int_{v_j}^{\tilde{c}_j} f_j(\tau_j)d\tau_j$ is the c.d.f. of $\tilde{c}_j$. In addition, the truthful payment to each node is given by $p_j(\tilde{\mathbf{c}}) = \tilde{c}_j \sum_{i\in\mathbf{R}_{sd}^{(j)}} x_i(\tilde{\mathbf{c}}) + \int_{\tilde{c}_j}^{w_j} \sum_{i\in\mathbf{R}_{sd}^{(j)}} x_i(\tau_j, \tilde{\mathbf{c}}_{-j})d\tau_j$, for $1 \leq j \leq n$.

The proof of Lemma 2 is provided in Appendix B. Note that the IC and IR constraints are now embodied in the objective function and the specific form of the truthful payment function presented in Lemma 2. According to Lemma 2, we can use the formulation in (7) to solve the simplified LPP problem. Let $h(\tilde{\mathbf{c}}) \overset{\text{def}}{=} \sum_{1\leq j\leq n} \left[\tilde{c}_j + \frac{F_j(\tilde{c}_j)}{f_j(\tilde{c}_j)}\right] \sum_{i\in\mathbf{R}_{sd}^{(j)}} x_i(\tilde{\mathbf{c}})$. An inspection of (7) shows that this formulation actually minimizes the expected value of $h(\tilde{\mathbf{c}})$, given the distribution $\mathbf{f}(\tilde{\mathbf{c}})$. A simple probabilistic argument says that the expected value will be minimized if $h(\tilde{\mathbf{c}})$ is minimized over every point of $\tilde{\mathbf{c}}$. This gives rise to the following lemma, which reduces our problem from a randomized setup to a deterministic one.

Lemma 3: Given the reported cost vector $\tilde{\mathbf{c}}$, the LPP problem can be expressed as follows:

$$\begin{array}{ll} \text{minimize} & \sum_{1\leq j\leq n} \left[\tilde{c}_j + \frac{F_j(\tilde{c}_j)}{f_j(\tilde{c}_j)}\right] \sum_{i\in\mathbf{R}_{sd}^{(j)}} x_i(\tilde{\mathbf{c}}) \\ s.t. & \\ & \sum_{i=1}^{N} x_i(\tilde{\mathbf{c}}) = 1 \end{array} \tag{8}$$

and the truthful payment is given by

$$p_j(\tilde{\mathbf{c}}) = \tilde{c}_j \sum_{i\in\mathbf{R}_{sd}^{(j)}} x_i(\tilde{\mathbf{c}}) + \int_{\tilde{c}_j}^{w_j} \sum_{i\in\mathbf{R}_{sd}^{(j)}} x_i(\tau_j, \tilde{\mathbf{c}}_{-j})d\tau_j, \quad 1 \leq j \leq n. \tag{9}$$

Regarding the solution of (8), we have the following results.

Theorem 1: In the simplified LPP problem, for a given $\tilde{\mathbf{c}}$, the optimal route contains only one path, which is given by

$$P_{sd}^{o}(\tilde{\mathbf{c}}) = \arg \min_{\forall P_{sd}^{(i)}\in\mathbf{R}_{sd}} \left(\sum_{j\in P_{sd}^{(i)}} \left[\tilde{c}_j + \frac{F_j(\tilde{c}_j)}{f_j(\tilde{c}_j)}\right] \right) \tag{10}$$

where $P_{sd}^{(i)}$ denotes the ith path in $\mathbf{R}_{sd}$, $1 \leq i \leq N$. Thus, the optimal routing vector is $\mathbf{X}^o(\tilde{\mathbf{c}}) = (0,\ldots,0,1,0,\ldots,0)$, where the non-zero element corresponds to the optimal path P_{sd}^{o}.

Proof: We rewrite $h(\tilde{\mathbf{c}})$ according to the routes in $\mathbf{R}_{sd}$:

$$h(\tilde{\mathbf{c}}) = \sum_{1\leq i\leq N} \left(\sum_{j\in P_{sd}^{(i)}} \left[\tilde{c}_j + \frac{F_j(\tilde{c}_j)}{f_j(\tilde{c}_j)}\right] \right) x_i(\tilde{\mathbf{c}}). \tag{11}$$

Let $\xi_j \overset{\text{def}}{=} \left[\tilde{c}_j + \frac{F_j(\tilde{c}_j)}{f_j(\tilde{c}_j)}\right]$. Note that for each j, $1 \leq j \leq N$, ξ_j is a constant for a given $\tilde{c}_j$. Accordingly, for each path i, the term $W_i \overset{\text{def}}{=} \left(\sum_{j\in P_{sd}^{(i)}} \xi_j\right)$ is also a constant. Therefore, for a given $\tilde{\mathbf{c}}$, problem (8) becomes a linear program (LP) of the form:

$$\begin{array}{ll} \text{minimize}_{(x_1,\ldots,x_N)} & \sum_{i=1}^{N} W_i x_i \\ s.t. & \\ & \sum_{i=1}^{N} x_i = 1. \end{array} \tag{12}$$

It is straightforward to see that the optimal solution to the above LP is $\mathbf{X}^o = (0,\ldots,0,1,0,\ldots,0)$, where the index of the non-zero element is $i^o = \arg\min_{1\leq i\leq N}\{W_i\}$. This proves Theorem 1. ■

Theorem 1 suggests the following algorithm for computing the LPP and the related payments. For a given $\tilde{\mathbf{c}}$, we use $\xi_j(\tilde{c}_j)$ as a weight for node j. The LPP is simply the shortest path from s to d w.r.t. ξ_j. If more than one path have the same shortest length, the algorithm breaks the tie by arbitrarily selecting one of them. Because ξ_j is a function of $\tilde{c}_j$, it may also be considered as the *virtual cost* of node j. Because now the optimal route contains only one path, the payment can be significantly simplified: Equation (9) shows that $p_j(\tilde{\mathbf{c}}) = 0$ if node j is not included in the shortest path. Otherwise, $p_j(\tilde{\mathbf{c}}) = \min(w_j, \bar{c}_j)$, where $\bar{c}_j$ is the "cutoff cost" of node j, beyond which node j will not be included in the LPP. This cutoff cost can be computed in the virtual cost domain. Specifically, in the virtual cost domain, let ζ

denote the difference in costs between the least-cost path that traverses node j and the least-cost path that does not traverse node j. If the virtual cost of node j increases from $\xi_j(\tilde{c}_j)$ to $\xi_j(\tilde{c}_j)+\zeta$, node j will not be included in the LPP. So $\bar{c}_j = \xi_j^{-1}(\zeta+\xi_j(\tilde{c}_j))$, where ξ_j^{-1} denotes the inverse of the function ξ_j. Note that because there is a $-\xi_j(\tilde{c}_j)$ term in ζ, $\zeta+\xi_j(\tilde{c}_j)$ is *independent* of $\xi_j(\tilde{c}_j)$ and thus is also independent of $\tilde{c}_j$. As a result, the payment received by node j is not related to $\tilde{c}_j$, and hence mis-reporting the cost of node j does not lead to greater profit for this node. Under the rational-user assumption, this means that an SR user will always report his true cost. A pseudo-code description of the above process is given in Algorithm 1.

Algorithm 1 Computing the LPP and payments without node-capacity constraints

```
INPUT: $\tilde{\mathbf{c}} = (\tilde{c}_1, \ldots, \tilde{c}_n)$
OUTPUT: $LPP$ and $payments$ to nodes
1:  for $j = 1$ to $n$ do
2:      $\xi_j \Leftarrow \tilde{c}_j + \frac{F_j(\tilde{c}_j)}{f_j(\tilde{c}_j)}$
3:  end for
4:  $LPP \Leftarrow$ shortest path from $s$ to $d$ w.r.t. weights $\xi_j$'s
5:  $minlength \Leftarrow$ length of($LPP$)
6:  $payments(j) \Leftarrow 0, \quad \forall j \notin LPP$
7:  for all nodes $j$ in $LPP$ do
8:      $length\_runnerup(j) \Leftarrow$ length of the shortest path that does not traverse node $j$
9:      $virtual\_cutoff\_cost(j) \Leftarrow length\_runnerup(j) - minlength + \xi_j$
10:     $payments(j) \Leftarrow \min\{\xi_j^{-1}(virtual\_cutoff\_cost(j)), w_j\}$
11: end for
```

Theorem 2: The route selection and payment mechanism given in Algorithm 1 is truthful and minimizes the expected price of the resulting route.

The validity of Theorem 2 is straightforward based on our previous discussion. In Algorithm 1, the computation of the LPP (line 4) involves finding a shortest path. Because all node weights are non-negative, this can be done using Dijkstra's algorithm in $\mathcal{O}(m + n\log n)$ time. It is easy to see that the worst-case running time of Algorithm 1 is $\mathcal{O}(mn + n^2\log n)$.

V. LPP WITH NODE-CAPACITY CONSTRAINTS

A. Optimal Route Selection and Truthful Payment Calculation

Based on (12), the inclusion of node-capacity constraints leads to the following formulation for the optimal route selection:

$$\begin{array}{ll}\text{minimize}_{\{x_1,\ldots,x_N\}} & \sum_{i=1}^{N} W_i x_i \\ s.t. & \\ & \sum_{i=1}^{N} x_i = 1 \\ & R\sum_{i\in\mathbf{R}_{sd}^{(j)}} x_i \le b_j, \forall j \in V.\end{array} \tag{13}$$

An inspection of (13) reveals that it is an LP. However, the main challenge in solving this problem using a conventional LP solver is that, in practice it is computationally infeasible to enumerate all the paths from s to d. So it is difficult to explicitly express the formulation (13). This hinders direct application of an LP solver.

Note that (13) belongs to the class of min-cost flow problems, but with non-negative nodal weights/capacities. So, instead of directly solving (13), we solve the following equivalent flow problem to minimize the same objective function:

1) Objective function:

$$\text{minimize}_{\{r_{jk}|\forall e_{jk}\in E,\, j\in V\backslash\{s,d\},\, k\in V\}} \sum_{j\in V\backslash\{s,d\}} \sum_{k\in V} \xi_j r_{jk} \tag{14}$$

2) Flow conservation constraint:

$$\sum_{k\in V} r_{kj} - \sum_{k\in V} r_{jk} = \begin{cases} -1, & j = s \\ 1, & j = d \\ 0, & \forall j \in V\backslash\{s,d\} \end{cases} \tag{15}$$

3) Node capacity constraint:

$$R\sum_{k\in V} r_{jk} \le b_j, \quad \forall j \in V\backslash\{s,d\} \tag{16}$$

where r_{jk} denotes the amount of traffic carried over a directed link e_{jk} from node j to node k, $e_{jk} \in E$ (if a directed link does not exist from node j to node k, then $r_{jk} = 0$). The flow optimization is w.r.t. r_{jk}'s. It is easy to see that the above flow-based optimization is equivalent to the optimization in (13): the flow formulation minimizes the total virtual cost of delivering traffic by optimizing the traffic carried by each link, while (13) minimizes the same objective by optimizing the traffic carried by each path. The optimization variables in these two formulations are related through the relationship that the traffic carried by a link is simply the summation of the traffic carried by all the paths passing through this link. So the optimal traffic allocation over paths in (13) also implies the optimal traffic allocation over links in the flow formulation. Compared with (13), the flow-based optimization (14) through (16) does not require enumerating all possible paths from the source to the destination. As a result, for a given graph G, the flow-based LP can be explicitly formulated. The LPP can be obtained by solving this LP using common LP solvers.

For the LPP with node-capacity constraints, the general form of the optimal payment defined in (9) still applies. This is because this general equation is solely decided by the IC and IR constraints, and is not related to whether the node-capacity constraint is present in the formulation. However, the challenge in applying this equation comes from the fact that (9) involves computing the integral of the traffic carried by a node j over various reported costs $\tilde{c}_j$. This integral requires evaluating the amount of traffic passing through node j at every value of $\tilde{c}_j$, as $\tilde{c}_j$ varies from the underlying reported value to the upper bound w_j. To exactly compute this integral, one must solve an infinite number of flow problems, one for each value of $\tilde{c}_j$, which is computationally infeasible. Furthermore, even if the relationship between the traffic carried by node j and its reported cost $\tilde{c}_j$ is explicitly known in the form of a function, say, $\pi(\tilde{c}_j)$, but the integration of π over $\tilde{c}_j$ is not known analytically, in general one still has to rely on numerical computations to calculate the approximate value of the integral.

B. Truthful Low-Priced Path (LOPP)

When node capacity constraints are present, the computational challenges associated with determining the payments under truthful LPP motivate us to consider a sub-optimal but computationally feasible version of the problem, which we refer to as truthful low-priced path (LOPP) routing. The price of a LOPP may not be the lowest, but is still significantly lower than the price of the truthful LCP. Keeping this setting in mind, we propose Algorithm 2 to solve the optimization problem in (13) approximately.

Algorithm 2 Computing a truthful LOPP under node-capacity constraints

```
INPUT: $\tilde{\mathbf{c}} = (\tilde{c}_1, \dots, \tilde{c}_n)$
OUTPUT: $LOPP$
1: for $j = 1$ to $n$ do
2:     $\xi_j \Leftarrow \tilde{c}_j + \frac{F_j(\tilde{c}_j)}{f_j(\tilde{c}_j)}$
3: end for
4: STEP 1: Find the shortest-path $P^*$ from $s$ to $d$ on the graph $G$ w.r.t. $\xi_j$'s.
5: STEP 2: Find the bottleneck node $j^*$ on path $P^*$. Allocate a fraction of traffic $b_{j^*}/R$ from the flow demand to this path. Update the capacity of each node on $P^*$ by subtracting $b_{j^*}$, the portion that has been used by the underlying shortest path, from its original capacity. Delete node $j^*$ from $G$, along with other possible bottleneck nodes whose capacities become 0.
6: STEP 3: Repeat STEPs 1 and 2 until all the traffic demand of the flow has been allocated or no additional shortest-path in STEP 1 can be found. In the former case, the sequence of paths found by the procedure along with the associated traffic allocation is the LOPP. If the latter case happens, then a LOPP cannot be found by the algorithm.
```

Algorithm 2 is equivalent to sorting the paths in $\mathbf{R}_{sd}$ increasingly according to their length W_i (the total virtual cost of the nodes on path i) and then allocating traffic to paths sequentially according to their ranking, until the traffic demand is satisfied. Each path is allocated an amount of traffic up to the capacity of its bottleneck node.

In general, the multi-path LOPP returned by Algorithm 2 is sub-optimal in terms of its price. For the special case where the routes under consideration are node-disjoint, Algorithm 2 actually finds the optimal solution to (13) (i.e., Algorithm 2 returns the LPP). For the general case, in each realization of Algorithm 2, the difference in the virtual cost of the path (i.e., the objective function in (13)) between the LOPP and the LPP can be evaluated by comparing the LOPP's total virtual cost with the result of the flow-based optimization (14). Although the LOPP found by Algorithm 2, in general, does not have the lowest price, our simulations show that, compared with the path found by the LCP algorithm, the average truthful price of the LOPP is significantly lower.

Algorithm 2 involves repetitive computation of several shortest paths. Because all weights are non-negative, Dijkstra's algorithm can be used to find the shortest path in each iteration in $\mathcal{O}(m + n \log n)$ time. It is straightforward to see that the worst-case running time of Algorithm 2 is $\mathcal{O}(mn + n^2 \log n)$.

C. Truthful Payment Calculation for LOPP

The traffic allocation vector computed by Algorithm 2, $(x_1^o, \dots, x_N^o)$, is an input to (9) to calculate node payments. The key challenge in calculating these payments is in computing the integral term in the equation. Note that in Algorithm 2, x_i^o is given only as an implicit function of $\tilde{c}_j$, so it is not

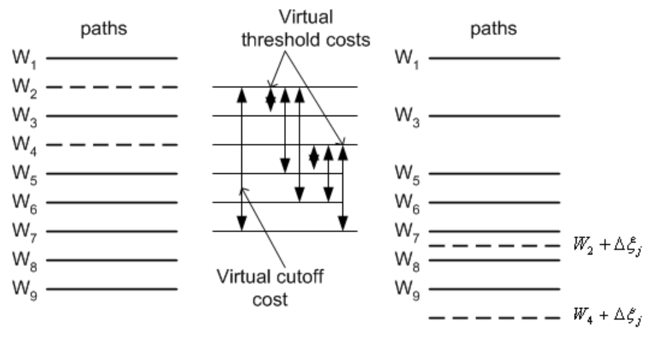


Fig. 1. Calculation of the virtual cutoff cost and threshold costs.

suitable to be used directly in the integration. To calculate the payments, we need to take a closer look at the relationship between node j's traffic and its reported cost $\tilde{c}_j$.

A re-examination of Algorithm 2 indicates that the traffic carried by a node j presents a multi-threshold structure in relation to $\tilde{c}_j$. More specifically, for a node j that is included in the LOPP route, the traffic it carries is the sum of the traffic carried by the paths that are part of this route and that also traverse node j. Let the set of these paths be $\mathbf{R}_{sd}^{(j)o}$. Algorithm 2 dictates that, given the capacity of each node, the optimal traffic allocation across various paths, i.e., $(x_1^o, \dots, x_N^o)$, is fully decided by the ranking of the paths according to their weights. As a result, when $\tilde{c}_j$ is increased, the traffic allocation to the paths in $\mathbf{R}_{sd}^{(j)o}$ will change only when the increment of $\tilde{c}_j$ is significant enough to change path ranking. It is not difficult to see that this is a mono-decreasing process, i.e., with an increase in $\tilde{c}_j$, the amount of traffic allocated to the paths in $\mathbf{R}_{sd}^{(j)o}$ always decreases after each change in ranking, because their ranking only drops with a larger $\tilde{c}_j$. Finally, no traffic will be allocated to the paths in $\mathbf{R}_{sd}^{(j)o}$, since all traffic has been allocated to the paths ranked before them. At this point, $\tilde{c}_j$ corresponds to the cutoff cost of the simplified LPP problem. The major difference here is that, before $\tilde{c}_j$ reaches the cutoff cost, there exist multiple threshold costs, at which a change in the ranking of the paths happens.

Such a multi-threshold structure largely simplifies our payment calculation, because the traffic allocation does not change between two consecutive threshold costs. So, to compute the integral in (9), we only need to find the threshold costs and evaluate the traffic allocation at these particular points.

We use an example to illustrate the basic idea of computing the threshold costs and cutoff cost for a node j that is part of the LOPP. Consider the example on the left of Figure 1, where paths in $\mathbf{R}_{sd}$ are labeled in the increasing order of their length W_i, defined based on the nodes' virtual costs. Suppose paths 1 to 5 constitute the multi-path LOPP, constructed according to Algorithm 2. Suppose paths 2 and 4 traverse node j. So, with an increase in ξ_j, the rankings of paths 2 and 4 drop simultaneously (note that the ranking of path 4 remains always below that of path 2, because $W_4 - W_2$ is constant, as shown in Figure 1). Now consider the situation shown on the right, where the ranking of path 2 has dropped to a position that no traffic will be allocated to it, but non-zero traffic is still allocated to the path ranked before it, say path 7 in this example. Note that at this time no traffic is allocated to path 4, either. The underlying increment in ξ_j, i.e., $\Delta\xi_j = W_7 - W_2$,

Algorithm 3 Payment calculation for LOPP under node-capacity constraints

```
INPUT: $LOPP$ and $\tilde{\mathbf{c}} = (\tilde{c}_1, \dots, \tilde{c}_n)$
OUTPUT: $payments$ to nodes
 1: for $j = 1$ to $n$ do
 2:     $\xi_j \Leftarrow \tilde{c}_j + \frac{F_j(\tilde{c}_j)}{f_j(\tilde{c}_j)}$
 3: end for
 4: for all $j \notin LOPP$ do
 5:     $payments(j) \Leftarrow 0$
 6: end for
 7: for all $j \in LOPP$ do
 8:     Construct $\mathbf{R}_{sd}^{(j)o}$ from $LPP$
 9:     Exclude node $j$ from $G$ and call Algorithm 2 to find the LOPP that does not include node $j$. $l_j^* \Leftarrow$ the length of the longest path in this LOPP that has non-zero traffic allocation
10:     $\mathbf{R}_{sd}(l_j^*) \Leftarrow \{$paths from $s$ to $d$ whose length is not longer than $l_j^*\} - \mathbf{R}_{sd}^{(j)o}$
11:     $k \Leftarrow 1$
12:     for each path $r \in \mathbf{R}_{sd}^{(j)o}$ and each path $u \in \mathbf{R}_{sd}(l_j^*)$ do
13:         threshold virtual cost $\xi_j^{(k)*} \Leftarrow \max\{$length of path $u$ − length of path $r, 0\} + \xi_j$
14:         threshold cost $\tilde{c}_j^{(k)*} \Leftarrow \min(\xi_j^{(-1)}(\xi_j^{(k)*}), w_j)$
15:         Call Algorithm 2 to evaluate the optimal traffic allocation $\mathbf{X}(\tilde{c}_j^{(k)*}, \tilde{\mathbf{c}}_{-j})$
16:         $k \Leftarrow k + 1$
17:     end for
18:     $K \Leftarrow$ number of threshold costs, including the cutoff cost
19:     Sort $\tilde{c}_j^{(k)*}$'s in an increasing order
20:     $payments(j) \Leftarrow \tilde{c}_j^{(1)*} \sum_{i \in \mathbf{R}_{sd}^{(j)}} x_i(\tilde{c}_j, \tilde{\mathbf{c}}_{-j}) + \sum_{k=1}^{K-1} \sum_{i \in \mathbf{R}_{sd}^{(j)}} x_i(\tilde{c}_j^{(k)*}, \tilde{\mathbf{c}}_{-j})(\tilde{c}_j^{(k+1)*} - \tilde{c}_j^{(k)*})$
21: end for
```

is the critical cutoff point for node j. If $\Delta\xi_j < W_7 - W_2$, traffic will still be allocated to path 2, so the traffic carried by node j will be non-zero. But if $\Delta\xi_j > W_7 - W_2$, no traffic passes through node j. So, the virtual cost of node j at this point, i.e., $\xi_j + (W_7 - W_2)$, is the node's cutoff virtual cost. A virtual threshold cost is simply the increment in ξ_j that leads to a change in the rankings of paths 2 or 4. As shown in the middle of Figure 1, in total there are 7 such increments: $W_5 - W_4, W_6 - W_4, W_7 - W_4, W_3 - W_2, W_5 - W_2, W_6 - W_2$, and $W_7 - W_2$ (the last corresponds to the cutoff virtual cost). The actual costs of node j can then be obtained from the corresponding virtual costs by the inverse mapping ξ_j^{-1}.

In the above calculations, the critical path 7 can be found by excluding node j from the graph G and calling Algorithm 2 to construct another LOPP that does not include node j. The longest path in this LOPP that has non-zero traffic allocation corresponds to the critical path 7 in the example. Note that there is no need to consider the paths after this critical path, because they will not receive any traffic allocation no matter how big ξ_j becomes. After getting the threshold costs and the cutoff cost, Algorithm 2 can be called to evaluate the traffic allocations at these particular costs. The above procedure for calculating the payments is formalized in Algorithm 3, which is executed after the LOPP is found by Algorithm 2.

The computational complexity of Algorithm 3 can be calculated as follows. For each node j that is included in the LOPP, the algorithm needs to call Algorithm 2 K times to evaluate the traffic allocation under the K threshold costs of node j. So the computation time for calculating the payment for one node is $\mathcal{O}(K(mn+n^2 \log n))$. The computation time for calculating the payments for the entire LOPP is $\mathcal{O}(Kmn^2 + Kn^3 \log n)$. In practice, the value of K has a big impact on the overall computational complexity. For each node j, K is at most the product of the number of paths in $\mathbf{R}_{sd}^{(j)o}$ and the paths in $\mathbf{R}_{sd}(l_j^*)$ (defined in line 10 of Algorithm 3). Intuitively, this implies that if the number of paths included in the LOPP is large, then K will be very large. This scenario happens when the source rate demand is much larger than a node capacity, such that the flow has to be split among several sub-flows.

D. Examples

We illustrate the operation of Algorithms 2 and 3 using the topology in Figure 2(a), where node s wants to send a flow of rate 1 to node d. The underlying reported cost and capacity of a node are shown in the figure in the form $(\tilde{c}_j, b_j)$. We assume that the cost of a node is uniformly distributed over $[0, 5]$. So the virtual cost is given by $\xi_j(\tilde{c}_j) = 2\tilde{c}_j$. The virtual costs of various nodes are shown in sub-figure (b). Algorithm 2 first returns path $s \to A \to B \to C \to d$ (length=3) as the shortest path. Because B and C are bottleneck nodes, 0.5 of the original flow is allocated to this path. Accordingly, the residual capacity of A is changed to $1 - 0.5 = 0.5$. Excluding nodes B and C, Algorithm 2 next determines path $s \to A \to E \to F \to d$ (length=4) as the shortest path. Now, A, E, and F are all bottleneck nodes. So 0.5 of the original flow is allocated to this path. Because the traffic demand has been fully allocated, Algorithm 2 terminates. So the LOPP contains two paths with the flow equally distributed between them.

Now consider how Algorithm 3 calculates the payments. Consider the payment to A, as an example. Both paths in LOPP traverse A, making it the most complicated case among all relaying nodes. To calculate the payment to A, Algorithm 3 first excludes A from the graph and calls Algorithm 2 to find an alternate LOPP that does not traverse A. This leads to the alternate 2-path LOPP: $s \to G \to E \to F \to d$ (weight=5, allocated traffic =0.5) and $s \to G \to H \to I \to d$ (weight=6, allocated traffic =0.5). The latter is the critical cutoff path. So there are 3 critical increments of ξ_A, namely $5 - 3 = 2$, $6 - 3 = 3$, $5 - 4 = 1$, and $6 - 4 = 2$. For $\Delta\xi_A = 1$, the new virtual cost of A is $1 + 1 = 2$. Beyond this new virtual cost, Algorithm 2 finds the new LOPP $s \to A \to B \to C \to d$ that carries 0.5 of the flow, and path $s \to G \to E \to F \to d$ that carries 0.5 of the flow. So overall A carries 0.5 of the flow. We repeat the above process and find that when $\Delta\xi_A = 2$, the LOPP consists of the two paths $s \to G \to E \to F \to d$ (traffic=0.5) and $s \to A \to B \to C \to d$ (traffic=0.5); when $\Delta\xi_A = 3$, the LOPP consists of $s \to G \to E \to F \to d$ (traffic=0.5) and $s \to G \to H \to I \to d$ (traffic=0.5). So the traffic of A is 0.5 and 0, respectively, when ξ_A goes beyond 3 and 4. The traffic of A vs. ξ_A is plotted in sub-figure (c). ξ_A is then inversely converted to the actual cost according to $\xi_A^{-1}(y) = 0.5y$, as shown in sub-figure (d). So the payment to A is $1 \times 0.5 + 1 \times (1 - 0.5) + 0.5 \times (2 - 1) = 1.5$.

E. Truthful Capacity Reporting

The above LOPP route selection and payment calculation procedure takes as input the capacity constraint b_j at each

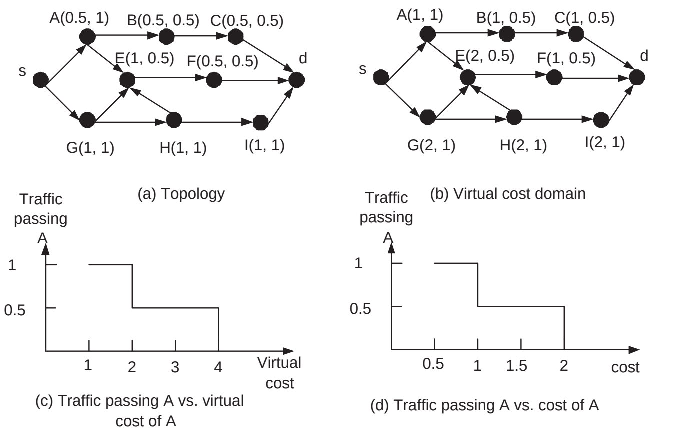


Fig. 2. Example of LOPP with a node-capacity constraint.

node. Just like the cost c_j, b_j should also be considered as private information that is only known to node j. As a result, node j may not report its true b_j to the routing mechanism, if that results in more profit for node j. For example, by exaggerating its capacity, node j may relay a larger fraction of traffic and receive a higher payment, leading to a possibly higher profit. To maintain the overall truthfulness of the mechanism, mis-reporting of capacity information should be eliminated by design.

We propose the following payment transaction algorithm that guarantees truthful capacity reporting by every node. At the beginning of the underlying session, the mechanism uses the reported node costs and capacities to compute the LOPP route $\mathbf{X}^o$ and the payment $\mathbf{p}^o$ based on Algorithms 2 and 3, respectively. After this calculation, s starts its transmission over route $\mathbf{X}^o$. The actual payment to intermediate nodes, a.k.a. payment transaction, will be deferred until the end of the underlying session. During the session, each intermediate node records the traffic volume it has carried. This traffic-recording is either based on the tamper-proof hardware proposed in [3] or on the cryptographic-receipt-based software proposed in [21]. Either way, it ensures that the actual traffic volume carried by a node is honestly recorded. After the session, source node s starts the payment transaction process. For SR node j, node s compares the volume of traffic that should be carried by j according to $\mathbf{X}^o$ with the actual volume of traffic that has been passed through j. The former quantity is calculated according to $\mathbf{X}^o$ as $s_j^o = \sum_{i\in\mathbf{R}_{sd}^{(j)}} x_i^o$. The latter one, denoted by s_j^*, comes from the traffic recording of node j. If $s_j^* < s_j^o$, then zero payment will be made to node j. Otherwise, the calculated payment p_j^o is paid to node j for every packet sent by s.

Theorem 3: The above payment materialization method results in truthful capacity reporting.

Proof: The basic idea of the proof is to show that if a node j can get a higher profit by mis-reporting b_j, then it must be a bottleneck node in the LOPP and its reported capacity must be greater than the actual b_j. Accordingly, the traffic allocated to this node must be greater than its capacity. So the actual traffic carried by node j must be smaller than its assigned traffic. The proposed method achieves truthfulness by specifically punishing this behavior. The detailed proof is as follows.

Denote the reported capacity of node j by $\tilde{b}_j$. Node j cheats by either reporting $\tilde{b}_j > b_j$ or $\tilde{b}_j < b_j$. Either way, the ranking of the paths from s to d in terms of length does not change with $\tilde{b}_j$. As a result, according to Algorithm 2, the order by which traffic demand is allocated to various paths does not change, but the traffic volume a path receives may change with $\tilde{b}_j$.

First note that node j does not have an incentive to under-report its capacity, i.e., claiming $\tilde{b}_j < b_j$. The reason is that by Algorithm 2, a smaller $\tilde{b}_j$ can only reduce or maintain, but never increase, the traffic carried by those paths that traverse through node j, i.e., $\sum_{i\in\mathbf{R}_{sd}^{(j)}} x_i(\tilde{\mathbf{c}})$ is non-decreasing with $\tilde{b}_j$. Substituting the payment (9) into (2), the profit of node j is given by

$$U_j(\mathbf{X},\mathbf{p},\tilde{c}_j) = \int_{\mathbf{D}_{-j}} \int_{\tilde{c}_j}^{w_j} \sum_{i\in\mathbf{R}_{sd}^{(j)}} x_i(\tau_j,\tilde{\mathbf{c}}_{-j}) d\tau_j \mathbf{f}_{-j}(\tilde{\mathbf{c}}_{-j}) d\tilde{\mathbf{c}}_{-j}. \tag{17}$$

Therefore, a smaller volume of traffic carried by node j only leads to smaller profit for node j. Thus node j does not have incentive to under-report its capacity.

Now consider the situation when $\tilde{b}_j > b_j$. Node j falls into one of the following four cases: (1) Node j is not part of the LOPP whether it reports b_j or $\tilde{b}_j$; (2) Node j is not part of the LOPP when reporting b_j, but is part of it when reporting $\tilde{b}_j$; (3) Node j is part of the LOPP when reporting b_j and is not part of it when reporting $\tilde{b}_j$; and (4) node j is part of the LOPP when reporting b_j and is also part of it when reporting $\tilde{b}_j$.

For case (1), the profit for node j is zero whether it reports b_j or $\tilde{b}_j$, so mis-reporting does not lead to a higher profit. Case (2) cannot happen for the following reason. If node j is not included in the LOPP when reporting b_j, then the ranking of the shortest path from s to d that traverses through node j is behind those paths that receive non-zero traffic allocation. This means that the paths that receive non-zero traffic allocation do not traverse node j. Therefore, increasing the reported capacity of node j does not change the order and volume by which these paths are allocated traffic. So node j cannot be included in the LOPP if it has not been included in it when reporting b_j.

In case (3), the profit of node j is greater than 0 when it reports b_j. When it reports $\tilde{b}_j$, it is not included in the route, so its profit is 0. So its profit is reduced when it mis-reports its capacity.

Case (4) contains two sub-cases. First, when b_j is reported, node j is not a bottleneck node of any path on the LOPP, and second, when b_j is reported, node j is a bottleneck node of some path that is part of the LOPP. For the first sub-case, reporting a larger capacity does not change the traffic allocation among routes, because such allocation is solely decided by bottleneck nodes. Increasing the capacity of node j does not change these bottleneck nodes, so the traffic allocation does not change. Similarly, reporting $\tilde{b}_j$ also does not change the calculated payment to node j, because the traffic allocation in the integral in (9) does not change. So node j's profit remains the same when the reported capacity is changed from b_j to $\tilde{b}_j$. For the second sub-case, because node j is a bottleneck node when b_j is reported, it saturates its capacity when relaying b_j amount of traffic. When $\tilde{b}_j$ is reported, more traffic will be allocated to node j according to Algorithm 2. This makes the traffic that is assigned to node j by Algorithm 2 exceed the node's actual capacity limit. As a result, during transmission, some traffic must be dropped at node j, making the actual relayed traffic volume smaller than the planned volume. According to the proposed payment materialization method, the actual payment to node j will be dropped to 0. Given that node j's profit is greater than 0 when b_j is reported, it is not in the interest of node j to report a higher capacity.

Combining all the above cases, Theorem 3 is proved. ■

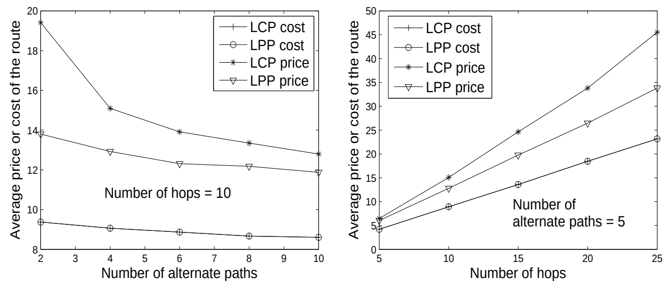
VI. Performance Evaluation

In this section, we evaluate the performance of LPP and LOPP using simulations. The VCG-based LCP algorithm is also studied. For an end-to-end flow, the computed cost and price are averaged over 500 sessions to obtain their expected value. Our simulations show that in general, similar performance trends are observed under different node cost distributions. So we only show the results under uniformly distributed node costs. The following results are averaged over 10 independent runs.

A. LPP Without Node-Capacity Constraints

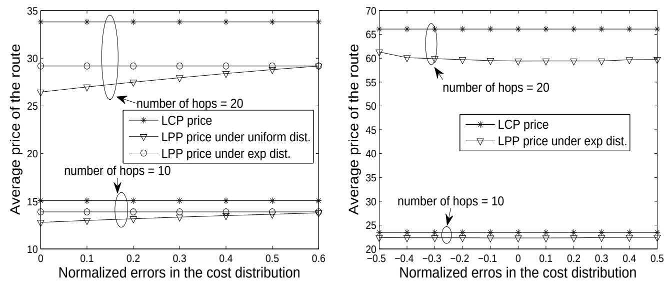
In Figure 3, we study the impact of network topology on performance. In these simulations, all paths from the source to the destination are node-disjoint and contain the same number of hops. Node costs are independently and uniformly distributed over $[0, 2]$. Albeit highly idealized, this topology allows us to directly control two important topological parameters: the number of alternate paths and their lengths. Figure 3 shows that a significant saving in price is achieved using LPP over LCP routing. At the same time, the cost difference between the two is trivial. This phenomenon indicates that LPP routing sacrifices little system-wide efficiency in order to achieve a lower payment for the end user. Two additional observations can be made. First, the saving in price is more significant when the number of alternate paths is small (sub-figure(a)). This is because under the VCG algorithm, the price of the LCP is the cost of the second-shortest path. When the total number of alternate paths is small, the monopoly effect of the second-shortest path becomes more significant, leading to a higher price for the LCP. Second, the saving due to LPP is more significant when the path length is larger (sub-figure(b)). This is due to the accumulative saving over various intermediate nodes along the path, which becomes more pronounced with an increase in the number of hops.

In Figure 4, we study the sensitivity of our algorithm to errors in estimating the distribution of the node cost. We are interested in two types of estimation errors: parameter error and distribution type error. Specifically, we generate the node cost according to a distribution $f^{(1)}$, but we assume the mechanism uses an estimated version, say $f^{(2)}$, to compute the virtual cost. For a parameter error, the distributions $f^{(1)}$ and $f^{(2)}$ are of the same type (function), but differ slightly



(a) Impact of no. of alternate paths (b) Impact of path length

Fig. 3. Impact of topology on the average price/cost of a path.



(a) Uniform (b) Exponential

Fig. 4. Sensitivity analysis under various cost distributions.

in the values of their parameters. We show the results for both uniformly and exponentially distributed node costs. For uniformly distributed costs, $f^{(1)}$ has a domain of $[v, w]$ and $f^{(2)}$ has a domain of $[v + 0.5\epsilon(w - v), w - 0.5\epsilon(w - v)]$, where $\epsilon > 0$ is the normalized error. For exponentially distributed costs, the rate parameters of $f^{(1)}$ and $f^{(2)}$ are given by $\lambda^{(2)} = (1 + \epsilon)\lambda^{(1)}$. For an error in the distribution type, $f^{(2)}$ and $f^{(1)}$ are different functions. In particular, when $f^{(1)}$ is uniform, we assume its estimated version, $f^{(2)}$, is an exponential distribution of the same mean. The price of the LPP under both estimation errors is plotted in Figure 4. From this figure, it is clear that as long as the means of $f^{(1)}$ and $f^{(2)}$ are close, both types of errors have only minor impact on the price of the LPP. This phenomenon indicates that our algorithm is insensitive to estimation errors in the node-cost distribution.

In Figure 5, we analyze the performance of LPP routing under random topologies. We consider a 1000 meter $\times 1000$ meter area. The source and destination are located at the middle of two opposite sides of the square, respectively. Other nodes are uniformly distributed. Heterogeneous spectrum opportunities are simulated. Specifically, we assume that in the middle of this square, there is a 200-meter-radius circular "hot" zone, in which the node cost is uniformly distributed in the range $[0.5, 5]$. For a node outside the hot zone, its cost is uniformly distributed in $[0.5, 2]$. Figure 5 shows that the price and cost of LPP and LCP decrease with the node's transmission range and with the number of nodes in the network. The savings in price due to LPP is more significant at small transmission ranges and small numbers of nodes. This can be explained by noting that a smaller transmission range corresponds to a longer path, and a smaller number of nodes means a smaller number of alternate paths from the source to the destination. So these trends are in line with our results in Figure 3. Sub-figures (c) and (d) show the percentage of sessions for which LPP$\neq$LCP. In general, in more than 10% of the sessions, the LPP differs from the LCP. Despite this large difference, sub-figures (a) and (b) show that the cost of the LPP is only slightly higher than the LCP. This phenomenon indicates that the LPP in general tends to employ a node of relatively low cost, but not necessarily the node of the lowest cost.

B. LOPP with Node-Capacity Constraints

1) A Single-Flow Scenario: In this set of simulation, we study LOPP routing under *static* traffic conditions. More specifically, our simulation is based on the same random topologies generated for Figure 5, but now each node is associated with a capacity limit (in Mbits/s). For a node in the hot zone, its capacity is randomly selected between 0 and 2 Mbits/s. Otherwise, the node capacity is selected between 0 and 5 Mbits/s. Once the node capacity is selected, it does not change throughout the simulation. We assume that there is only one flow inside the network. The source and destination are fixed at the middle of the two oppositive sides of the simulated square area, respectively. The flow has a constant rate demand for all its sessions. Due to the lack of counterpart multi-path LCP algorithms, we compare the multi-path route found by LOPP, denoted as MLPP in our results, with two single-path routing algorithms. These two algorithms are straightforward extensions of the VCG-LCP and the simplified LPP (the one that does not consider the node capacity constraint), where we prune out from the topology nodes whose capacities are smaller than the rate demand. The LCP and the simplified LPP algorithms are then applied to the residual topology. To distinguish it from the MLPP, the extension of the simplified LPP is denoted as SLPP.

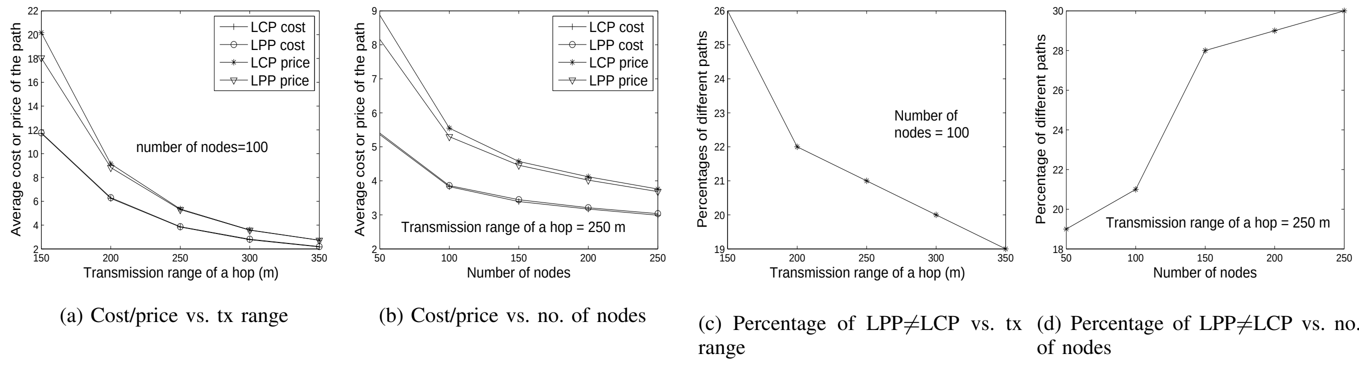
In Figure 6(a) we plot the average price/cost of various mechanisms as a function of the rate demand. It is clear that the cost and price of MLPP are significantly smaller than its single-path counterparts. In addition, the figure shows that one problem of single-path routing algorithms is that the destination may become unreachable after pruning out nodes of low capacity. In contrast, MLPP supports a much higher rate demand by “bonding” the capacities of multiple paths. This point is clear in sub-figure (b), where the average number of paths for the MLPP is plotted as a function of the rate demand. It is evident that as we increase the rate demand, more paths are included in the MLPP.

In Figure 6(c), we plot the average price/cost of various mechanisms as a function of the number of nodes. The figure shows that the cost and price decrease with the number of nodes. This can be explained by noting that with higher node density, more alternative paths become available between the source and the destination, which reduces the cost and price of the computed route. The average number of paths in the MLPP is plotted as a function of the number of nodes in Figure 6(d). Initially, the number of paths in the MLPP increases with the number of nodes, but once the number of nodes becomes sufficiently large, the number of paths in the MLPP begins to decrease with node density. This phenomenon can be explained as follows. At low node densities, the number of nodes that simultaneously have small costs and high capacities is small. As node density increases, the cost of a path may become smaller, but the capacity of its bottleneck node may not improve. This causes traffic to be distributed among more paths that are of lower cost. Once the node density is sufficiently high, the number of nodes that have small cost and also high capacity becomes sufficiently large such that a path of low cost can also accommodate more traffic, which reduces the number of paths in the MLPP route.

2) A Multi-flow Scenario: In this set of simulations, we are interested in the performance of the proposed algorithms under *dynamic* traffic conditions. We assume that there are multiple flows in the network, each of which alternates between ON and OFF periods. An ON period corresponds to an active session. No traffic is generated during the OFF period. We assume that the lengths of the ON and OFF periods follow an exponential distribution with means $T_{ON} = 20$ seconds and $T_{OFF} = 20$ seconds. Capacity allocation at intermediate nodes is performed on a dynamic basis: capacity is allocated to a flow during the ON periods and returned to the node during the OFF periods. So, the main difference between this scenario and the one assumed in the previous section is that, in here, the instantaneous residual capacity at an intermediate node fluctuates randomly with the real-time traffic conditions. As a result, some sessions of a flow may be denied a route due to insufficient instantaneous residual capacity at some intermediate nodes. We are interested in the blocking probability of the flows, i.e., the ratio of sessions for which a MLPP/SLPP/LCP route could not be found, and thus the session is denied admission to the network. We consider the same random topology and node cost settings as in the previous simulations. The capacity of each node is decided in the same way as in the single-flow scenario. We assume that the rate demand of a flow is uniformly sampled between 0 and 5 Mbps, and does not change during the simulation. The source and destination of a flow are randomly selected and are fixed in each simulation run. We simulate 10000 seconds, corresponding to an average of 250 sessions for each flow.

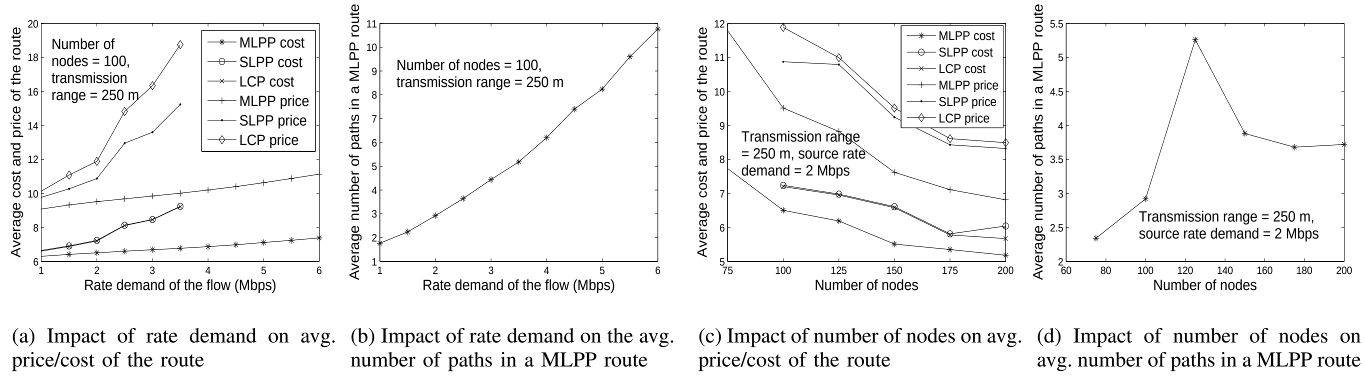
The blocking probability under various mechanisms is plotted as a function of the number of nodes in Figure 7(a). It can be observed that the blocking probability under MLPP is much lower than its single-path counterpart. In addition, as we increase the number of nodes, the blocking probability under MLPP is reduced quickly, because those newly added nodes begin to carry traffic. We plot the average number of paths contained in the MLPP versus the number of nodes in Figure 7(b). For the same reason explained in the single-flow case, a similar trend is observed as those in Figure 6(d).

In Figure 7(c), we plot the blocking probability as a function of the number of flows. Consistently, the MLPP is shown to be more accommodative of traffic dynamics than its single-path counterparts. The average number of paths contained in the MLPP versus the number of flows is plotted in Figure 7(d). It is shown that MLPP starts to include more paths with an increase in the number of flows, because of the increased traffic. However, when the number of flows is too large,



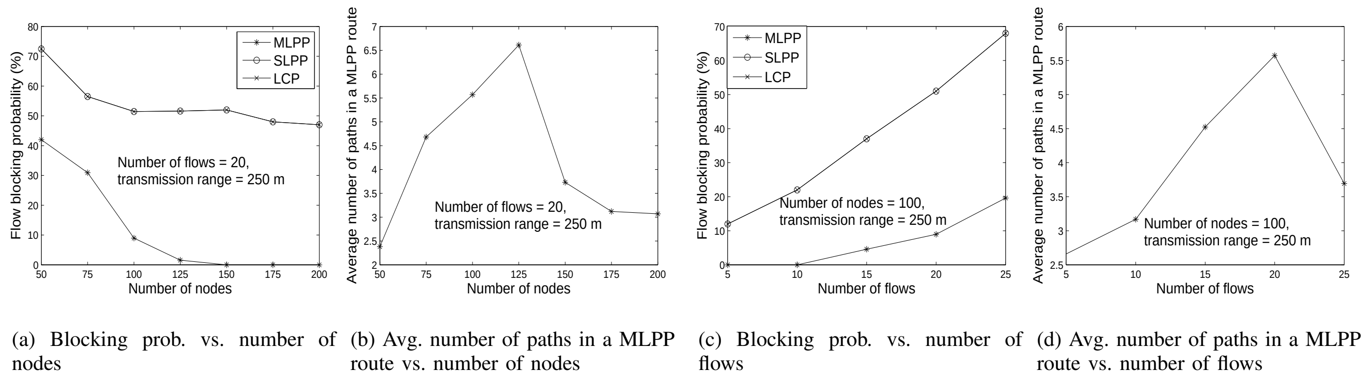
(a) Cost/price vs. tx range (b) Cost/price vs. no. of nodes (c) Percentage of LPP≠LCP vs. tx range (d) Percentage of LPP≠LCP vs. no. of nodes

Fig. 5. LPP under random topologies.



(a) Impact of rate demand on avg. price/cost of the route (b) Impact of rate demand on the avg. number of paths in a MLPP route (c) Impact of number of nodes on avg. price/cost of the route (d) Impact of number of nodes on avg. number of paths in a MLPP route

Fig. 6. Impact of static traffic condition on route selection (single-flow case).



(a) Blocking prob. vs. number of nodes (b) Avg. number of paths in a MLPP route vs. number of nodes (c) Blocking prob. vs. number of flows (d) Avg. number of paths in a MLPP route vs. number of flows

Fig. 7. Impact of dynamic traffic conditions on route selection (multi-flow case).

the number of paths contained in the MLPP begins to drop, because now more traffic is blocked.

VII. Conclusions

We studied the truthful LPP routing in an OSA network without and with node-capacity constraints. In the former case, we introduced polynomial-time algorithms for finding the LPP and calculating the payments to SR nodes. In the latter case, we derived the analytical solution for the LPP and its truthful payment, but pointed out that the calculation of the truthful payment is, in general, computationally infeasible. This finding motivated us to study a sub-optimal but more computationally feasible version of the problem, namely, the truthful LOPP route, which we addressed by proposing a polynomial-time algorithm. Compared with LCP, the adoption of LPP (or LOPP when there are node-capacity constraints) can lower the price paid by end users. The saving in payment using LPP and LOPP largely depends on the number of alternate paths between the source and the destination, and the length (in number of hops) of these paths. In general, the more alternate paths and the longer these paths are, the more saving the LPP and LOPP can achieve. Although the LPP and LOPP result in a lower price tag for end users, its cost is only slightly higher than that of the LCP. This indicates that the LPP and LOPP mechanisms only need to sacrifice a trivial amount of social efficiency in exchange for a lower price tag. In addition, our study has also shown that when capacity constraint are present, the proposed LOPP algorithm can accommodate more traffic in an economical way than the LCP algorithm, by intelligently splitting a flow into multiple sub-flows and establishing paths for each sub-flow.

Some open topics remain for future research. In this paper, we have focused on the mathematical structures of the LPP mechanism. Protocols that take into account practical issues related to its implementation are yet to be developed. In addition, collusion between nodes is not considered in this work. This will be studied in a future work.

Acknowledgements

This research was supported in part by NSF (under grants CNS-1016943, CNS-0721935, CNS-0904681, and IIP-0832238), Raytheon, and the "Connection One" Center. Any opinions, findings, conclusions, or recommendations expressed in this paper are those of the author(s) and do not necessarily reflect the views of the NSF. The authors would like to thank Professor Srini Ramasubramanian for his insightful comments regarding the technical content of this work.

References

[1] 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 ACM MobiCom Conference*, 2003.
[2] A. Archer and E. Tardos. Frugal path mechanisms. In *Proceedings of the Anunal ACM SIAM Symposium on Discrete Algorithms (SODA)*, 2002.
[3] L. Buttyan and J. P. Hubaux. Enforcing service availability in mobile ad-hoc WANs. In *Proceedings of the ACM MobiHoc Conference*, 2000.
[4] E. Elkind, L. A. Goldberg, and P. W. Goldberg. Frugality ratios and improved truthful mechanism for vertex cover. *Technical Report, University of Southampton, available at http://arxiv.org/abs/cs/0606044v4*, 2008.
[5] E. Elkind, A. Sahai, and K. Steiglitz. Frugality in path auctions. In *Proceedings of the Anunal ACM SIAM Symposium on Discrete Algorithms (SODA)*, 2004.
[6] J. Feigenbaum, C. Papadimitriou, R. Sami, and S. Shenker. A BGP-based mechanism for lowest-cost routing. *Distributed Computing Journal*, 18:61–72, 2005.
[7] J. Hershberger and S. Suri. Vickrey prices and shortest paths: what is an edge worth? In *Proceedings of the IEEE Symposium on Foundations of Computer Science*, pages 252–259, 2001.
[8] J. Jia, Q. Zhang, Q. Zhang, and M. Liu. Revenue generation for truthful spectrum auction in dynamic spectrum access. In *Proceedings of the ACM MobiHoc Conference*, 2009.
[9] X. Y. Li, Y. Wu, P. Xu, G. Chen, and M. Li. Hidden information and actions in multi-hop wireless ad hoc networks. In *Proceedings of the ACM MobiHoc Conference*, pages 283–292, 2008.
[10] A. H. Mohsenian-Rad, V. W. S. Wong, and V. C. M. Leung. Two-fold pricing to guarantee individual profits and maximum social welfare in multi-hop wireless access networks. *IEEE Transactions on Wireless Communications*, 8(8):4110–4121, Aug. 2009.
[11] R. B. Myerson. Optimal auction design. *Mathematics of Operations Research*, 6(1):58–73, Feb. 1981.
[12] M. Neely. Optimal pricing in a free market wireless network. *Elsevier Journal of Computer Networks*, 15:901–915, 2009.
[13] N. Nisan and A. Ronen. Algorithmic mechanism design. *Games and Economic Behavior*, 35:166–196, 2001.
[14] T. Shu and M. Krunz. Exploiting microscopic spectrum opportunities in cognitive radio networks via coordinated channel access. *IEEE Transactions on Mobile Computing*, 9(11):1522–1534, 2010.
[15] X. Su, S. Chan, and G. Peng. Auction in multi-path multi-hop routing. *IEEE Communication Letter*, 13(2):154–156, Feb. 2009.
[16] W. Wang, S. Eidenbenz, Y. Wang, and X. Y. Li. OURS: optimal unicast routing systems in non-cooperative wireless networks. In *Proceedings of the ACM MobiCom Conference*, 2006.
[17] W. Wang and X. Y. Li. Low-cost routing in selfish and rational wireless ad hoc networks. *IEEE Transactions on Mobile Computing*, 5(5):596–607, May 2006.
[18] W. Wang, X. Y. Li, and Y. Wang. Truthful multicast routing in selfish wireless networks. In *Proceedings of the ACM MobiCom Conference*, pages 245–259, 2004.
[19] F. Wu, T. Chen, S. Zhong, L. Li, and Y. R. Yang. Incentive-compatible opportunistic routing for wireless networks. In *Proceedings of the ACM MobiCom Conference*, pages 303–314, 2008.
[20] Q. Zhao and B. M. Sadler. A survey of dynamic spectrum access: signal processing, networking, and regulatory policy. *IEEE Signal Processing Magazine*, 24(3):79–89, 2007.
[21] 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 IEEE INFOCOM Conference*, 2003.
[22] S. Zhong, L. Li, Y. G. Liu, and Y. R. Yang. On designing incentive-compatible routing and forwarding protocols in wireless ad-hoc networks–an integrated approach using game theoretical and cryptographic techniques. In *Proceedings of the ACM MobiCom Conference*, 2005.
[23] X. Zhou, S. Gandhi, S. Suri, and H. Zheng. eBay in the sky: strategy-proof wireless spectrum auctions. In *Proceedings of the ACM MobiCom Conference*, 2008.

Appendix A
Proof of Lemma 1

Our proof follows the same procedure in [11]. Given two distinct numbers $\tilde{c}_j^{(1)} \in D_j$ and $\tilde{c}_j^{(2)} \in D_j$, the inequality in the IC constraint (2) implies that

$$U_j(\mathbf{X}, \mathbf{p}, \tilde{c}_j^{(1)}) \geq \int_{\mathbf{D}_{-j}} \left(p_j(\tilde{c}_j^{(2)}, \tilde{\mathbf{c}}_{-j}) - \tilde{c}_j^{(1)} \sum_{i \in \mathbf{R}_{sd}^{(j)}} x_i(\tilde{c}_j^{(2)}, \tilde{\mathbf{c}}_{-j}) \right) \times \mathbf{f}_{-j}(\tilde{\mathbf{c}}_{-j}) d\tilde{\mathbf{c}}_{-j} \quad (18)$$

$$U_j(\mathbf{X}, \mathbf{p}, \tilde{c}_j^{(2)}) \geq \int_{\mathbf{D}_{-j}} \left(p_j(\tilde{c}_j^{(1)}, \tilde{\mathbf{c}}_{-j}) - \tilde{c}_j^{(2)} \sum_{i \in \mathbf{R}_{sd}^{(j)}} x_i(\tilde{c}_j^{(1)}, \tilde{\mathbf{c}}_{-j}) \right) \times \mathbf{f}_{-j}(\tilde{\mathbf{c}}_{-j}) d\tilde{\mathbf{c}}_{-j}. \quad (19)$$

In addition, we can show the following

$$\begin{aligned} & \int_{\mathbf{D}_{-j}} \left[p_j(\tilde{c}_j^{(2)}, \tilde{\mathbf{c}}_{-j}) - \tilde{c}_j^{(1)} \sum_{i \in \mathbf{R}_{sd}^{(j)}} x_i(\tilde{c}_j^{(2)}, \tilde{\mathbf{c}}_{-j}) \right] \mathbf{f}_{-j}(\tilde{\mathbf{c}}_{-j}) d\tilde{\mathbf{c}}_{-j} \\ = & \int_{\mathbf{D}_{-j}} \left[p_j(\tilde{c}_j^{(2)}, \tilde{\mathbf{c}}_{-j}) - \tilde{c}_j^{(2)} \sum_{i \in \mathbf{R}_{sd}^{(j)}} x_i(\tilde{c}_j^{(2)}, \tilde{\mathbf{c}}_{-j}) \right] \mathbf{f}_{-j}(\tilde{\mathbf{c}}_{-j}) d\tilde{\mathbf{c}}_{-j} \\ & - \int_{\mathbf{D}_{-j}} (\tilde{c}_j^{(1)} - \tilde{c}_j^{(2)}) \sum_{i \in \mathbf{R}_{sd}^{(j)}} x_i(\tilde{c}_j^{(2)}, \tilde{\mathbf{c}}_{-j}) \mathbf{f}_{-j}(\tilde{\mathbf{c}}_{-j}) d\tilde{\mathbf{c}}_{-j} \\ = & \; U_j(\mathbf{X}, \mathbf{p}, \tilde{c}_j^{(2)}) - (\tilde{c}_j^{(1)} - \tilde{c}_j^{(2)}) Q_j(\tilde{c}_j^{(2)}). \quad (20) \end{aligned}$$

Substituting (20) to (18), we get

$$U_j(\mathbf{X}, \mathbf{p}, \tilde{c}_j^{(1)}) \geq U_j(\mathbf{X}, \mathbf{p}, \tilde{c}_j^{(2)}) - (\tilde{c}_j^{(1)} - \tilde{c}_j^{(2)}) Q_j(\tilde{c}_j^{(2)}). \quad (21)$$

Following the same treatment, but now consider (19) we get

$$U_j(\mathbf{X}, \mathbf{p}, \tilde{c}_j^{(2)}) \geq U_j(\mathbf{X}, \mathbf{p}, \tilde{c}_j^{(1)}) - (\tilde{c}_j^{(2)} - \tilde{c}_j^{(1)}) Q_j(\tilde{c}_j^{(1)}). \quad (22)$$

Combining (21) and (22), we derive

$$(\tilde{c}_j^{(2)} - \tilde{c}_j^{(1)}) Q_j(\tilde{c}_j^{(2)}) \leq U_j(\mathbf{X}, \mathbf{p}, \tilde{c}_j^{(1)}) - U_j(\mathbf{X}, \mathbf{p}, \tilde{c}_j^{(2)}) \leq (\tilde{c}_j^{(2)} - \tilde{c}_j^{(1)}) Q_j(\tilde{c}_j^{(1)}). \quad (23)$$

This means $(\tilde{c}_j^{(2)} - \tilde{c}_j^{(1)}) Q_j(\tilde{c}_j^{(2)}) \leq (\tilde{c}_j^{(2)} - \tilde{c}_j^{(1)}) Q_j(\tilde{c}_j^{(1)})$. So condition (1) of Lemma 1 follows.

Note that the inequalities in (23) can be rewritten for any $\delta > 0$ as

$$\delta Q_j(\tilde{c}_j^{(2)}) \leq U_j(\mathbf{X}, \mathbf{p}, \tilde{c}_j^{(2)} - \delta) - U_j(\mathbf{X}, \mathbf{p}, \tilde{c}_j^{(2)}) \leq \delta Q_j(\tilde{c}_j^{(2)} - \delta). \quad (24)$$

Since $Q_j(\tilde{c}_j)$ is decreasing in $\tilde{c}_j$, it is Riemann integrable [11]. So

$$\int_{\tilde{c}_j}^{w_j} Q_j(\tau_j) d\tau_j = U_j(\mathbf{X}, \mathbf{p}, \tilde{c}_j) - U_j(\mathbf{X}, \mathbf{p}, w_j). \quad (25)$$

This proves condition 2 of Lemma 1. ■

Appendix B
Proof of Lemma 2

The proof follows the method used in [11]. In particular, the objective function in (1) can be rewritten as follows

$$\begin{aligned} \Gamma_s(\mathbf{X}, \mathbf{p}) = & \sum_{1 \leq j \leq n} \int_{\mathbf{D}} \tilde{c}_j \sum_{i \in \mathbf{R}_{sd}^{(j)}} x_i(\tilde{\mathbf{c}}) \mathbf{f}(\tilde{\mathbf{c}}) d\tilde{\mathbf{c}} \\ & + \sum_{1 \leq j \leq n} \int_{\mathbf{D}} \left[p_j(\tilde{\mathbf{c}}) - \tilde{c}_j \sum_{i \in \mathbf{R}_{sd}^{(j)}} x_i(\tilde{\mathbf{c}}) \right] \mathbf{f}(\tilde{\mathbf{c}}) d\tilde{\mathbf{c}}. \quad (26) \end{aligned}$$

The second term in (26) can be written as

$$\begin{aligned} & \int_{\mathbf{D}} \left[p_j(\tilde{\mathbf{c}}) - \tilde{c}_j \sum_{i \in \mathbf{R}_{sd}^{(j)}} x_i(\tilde{\mathbf{c}}) \right] \mathbf{f}(\tilde{\mathbf{c}}) d\tilde{\mathbf{c}} \\ = & \int_{v_j}^{w_j} U_j(\mathbf{X}, \mathbf{p}, \tilde{c}_j) \mathbf{f}_j(\tilde{c}_j) d\tilde{c}_j. \quad (27) \end{aligned}$$

Substituting condition (2) of Lemma 1 into (27), we get

$$\begin{aligned} & \int_{\mathbf{D}} \left[p_j(\tilde{\mathbf{c}}) - \tilde{c}_j \sum_{i \in \mathbf{R}_{sd}^{(j)}} x_i(\tilde{\mathbf{c}}) \right] \mathbf{f}(\tilde{\mathbf{c}}) d\tilde{\mathbf{c}} \\ = & \; U_j(\mathbf{X}, \mathbf{p}, w_j) + \int_{v_j}^{w_j} \int_{\tilde{c}_j}^{w_j} Q_j(\tau_j) d\tau_j f_j(\tilde{c}_j) d\tilde{c}_j \\ = & \; U_j(\mathbf{X}, \mathbf{p}, w_j) + \int_{\mathbf{D}} F_j(\tilde{c}_j) \sum_{i \in \mathbf{R}_{sd}^{(j)}} x_i(\tilde{\mathbf{c}}) \mathbf{f}_{-j}(\tilde{\mathbf{c}}_{-j}) d\mathbf{c}. \quad (28) \end{aligned}$$

Substituting (28) into (26), after some mathematical manipulation, the objective function of (1) can be rewritten as

$$\begin{aligned} \Gamma_s(\mathbf{X}, \mathbf{p}) = & \sum_{1 \leq j \leq n} \int_{\mathbf{D}} \left[\tilde{c}_j + \frac{F_j(\tilde{c}_j)}{f_j(\tilde{c}_j)} \right] \sum_{i \in \mathbf{R}_{sd}^{(j)}} x_i(\tilde{\mathbf{c}}) \mathbf{f}(\tilde{\mathbf{c}}) d\tilde{\mathbf{c}} \\ & + \sum_{1 \leq j \leq n} U_j(\mathbf{X}, \mathbf{p}, w_j). \quad (29) \end{aligned}$$

It is easy to show that when the payment is $p_j(\tilde{\mathbf{c}}) = \tilde{c}_j \sum_{i \in \mathbf{R}_{sd}^{(j)}} x_i(\tilde{\mathbf{c}}) + \int_{\tilde{c}_j}^{w_j} \sum_{i \in \mathbf{R}_{sd}^{(j)}} x_i(\tau_j, \tilde{\mathbf{c}}_{-j}) d\tau_j$, for $1 \leq j \leq n$, the second term in (29), i.e., $U_j(\mathbf{X}, \mathbf{p}, w_j)$, equals 0. According to the Individual Rationality constraint, $U_j \geq 0$. Therefore, the payment method in Lemma 2 minimizes $U_j(\mathbf{X}, \mathbf{p}, w_j)$. As a result, the objective of minimizing $\Gamma_s(\mathbf{X}, \mathbf{p})$ is reduced to minimizing the new objective function $\sum_{1 \leq j \leq n} \int_{\mathbf{D}} \left[\tilde{c}_j + \frac{F_j(\tilde{c}_j)}{f_j(\tilde{c}_j)} \right] \sum_{i \in \mathbf{R}_{sd}^{(j)}} x_i(\tilde{\mathbf{c}}) \mathbf{f}(\tilde{\mathbf{c}}) d\tilde{\mathbf{c}}$. It is easy to verify that the payment function always satisfies the IR constraint, because $p_j(\tilde{\mathbf{c}}) \geq \tilde{c}_j \sum_{i \in \mathbf{R}_{sd}^{(j)}} x_i(\tilde{\mathbf{c}})$. The only constraint which has not been reflected is the multi-path constraint, which we append to the new optimization formulation. This proves Lemma 2. ■

PLACE PHOTO HERE

Tao Shu received his Ph.D. degree in Electrical and Computer Engineering from The University of Arizona in Dec. 2010. He received his B.S. and M.S. degrees in Electronic Engineering from the South China University of Technology, Guangzhou, China, in 1996 and 1999, respectively, and a Ph.D. degree in Communication and Information Systems from Tsinghua University, Beijing, China, in 2003. From Dec. 2010 to July 2011, Dr. Shu was a senior engineer at Qualcomm Inc., San Jose, CA. Since August 2011, he has been an assistant professor in the Department of Computer Science and Engineering, Oakland University. Dr. Shu's research aims at addressing security and performance issues in wireless networking systems, with strong emphasis on system architecture, protocol design, and performance optimization.

PLACE PHOTO HERE

Marwan Krunz is a professor of ECE at the University of Arizona. He also holds a joint appointment at the same rank in the Department of Computer Science. He is also the UA site director for Connection One, a joint NSF/state/industry IUCRC cooperative center that focuses on RF and wireless communication systems and networks. Dr. Krunz received his Ph.D. degree in electrical engineering from Michigan State University in 1995. He joined the University of Arizona in January 1997, after a brief postdoctoral stint at the University of Maryland, College Park. During the summer of 2011, he visited with the University of Jordan, King Abdullah II School of Information Technology, as a Fulbright Senior Specialist. In 2010, he was on sabbatical leave at the University of Carlos III de Madrid (Spain), where he held a Visiting Chair of Excellence ("Cátedra de Excelencia") position. Concurrently, he was a visiting researcher at Institute IMDEA, Madrid. He previously held other visiting positions at INRIA (Sophia Antipolis, France), HP Labs (Palo Alto, California), University of Paris VI (Paris, France), and US West (now Qwest) Advanced Technologies (Boulder, Colorado).

Dr. Krunz's research interests lie in the fields of computer networking and wireless communications, with recent focus on cognitive radios and SDRs; distributed radio resource management in wireless networks; channel access and protocol design; MIMO and smart-antenna systems; UWB-based personal area networks; energy management and clustering in sensor networks; media streaming; QoS routing; and fault monitoring/detection in optical networks. He has published more than 170 journal articles and refereed conference papers, and is a co-inventor on three US patents. M. Krunz is a recipient of the NSF CAREER Award (1998). He served on the editorial boards for the IEEE/ACM Transactions on Networking (2001–2008), the IEEE Transactions on Mobile Computing (2006–2011), and the Computer Communications Journal (2001–2011). He currently serves on the editorial board for the IEEE Transactions on Network and Service Management (TNSM). He was a guest co-editor for special issues in IEEE Micro and IEEE Communications magazines. He served as a technical program chair for various IEEE and non-IEEE conferences, including WoWMoM'06, SECON'05, INFOCOM'04, and the 2001 Hot Interconnects Symposium. He has served and continues to serve on the executive and technical program committees of many international conferences and on the panels of several NSF directorates. He gave keynotes and tutorials, and participated in various panels at premier wireless networking conferences. He is a consultant for a number of companies in the telecommunications sector. He is an IEEE Fellow.