6-14-2009

Identifier-based discovery mechanism design in large-scale networks

Chaouki T. Abdallah

Joud Khoury

Follow this and additional works at: https://digitalrepository.unm.edu/ece_fsp

Recommended Citation

Abdallah, Chaouki T. and Joud Khoury. "Identifier-based discovery mechanism design in large-scale networks." *IEEE International Conference on Communications Workshops* (2009): 1-6. doi:10.1109/ICCW.2009.5207997.

This Article is brought to you for free and open access by the Engineering Publications at UNM Digital Repository. It has been accepted for inclusion in Electrical & Computer Engineering Faculty Publications by an authorized administrator of UNM Digital Repository. For more information, please contact disc@unm.edu.

Identifier-based Discovery Mechanism Design in Large-Scale Networks

Joud Khoury and Chaouki T. Abdallah

1 University of New Mexico, Albuquerque NM 87131

{jkhoury, chaouki}@ece.unm.edu

Abstract—This paper introduces identifier-based discovery mechanisms in large-scale networks and presents a taxonomy of discovery schemes and proposals based on their business model. The paper poses several questions that are becoming increasingly important as we proceed to design the inter-network of the future. We first define and motivate a multi-level discovery problem that accounts for discovery service differentiation and we present an architecture that provides differentiated services. We then proceed to present an economic framework that accounts for the cost of discovery and for the players' valuations and to discuss the inherent challenges.

I. INTRODUCTION AND MOTIVATION

Almost every networking application relies on discovery and naming/identification services. An identifier in this context refers to an address that is independent of the network topology but that could nevertheless be "routable"¹. Identifier-based discovery (simply referred to as discovery hereafter) is a core network service aimed at discovering a network path to an identified object. Discovery is usually the first step in communication, even before a path to the destination object is established. Given an identifier of some object on the network, discovering a path to the object could either utilize mapping/resolution where the identifier is mapped to some locator² (see for example [1], [2], [3], and the Domain Name System (DNS)), or it could utilize routing-on-identifiers (see [4], [5], [6], [7] etc.). In either case however, an underlying routing scheme that routes on locators typically exists and is utilized after a path has been discovered for efficient communication. Note that the terms identifier and locator are both addresses at different layers of abstraction. We differentiate the two terms only after we fix an upper layer: an identifier at the upper layer maps into a locator which is an address relative to the upper layer. The locator itself is a path identifier at a lower layer.

This paper is concerned specifically with the design of discovery mechanisms. A named object (such as a node or service), referred to as a player, demands to be discoverable by the rest of the network. A discovery mechanism provides such service to the players. We define the *discovery level* to be a measure of "how discoverable" a player is by the rest of the network³. The performance of discovery, or the discovery

level, could significantly affect the player's business model especially in time-sensitive application contexts. If discovering an object takes a significant time relative to the object's download time, the requesting user's experience suffers. As an example, when no caching is involved, the DNS resolution latency comprises a significant part of the total latency to download a webpage (10-30 %) [8], [9]. This overhead becomes more noticeable in Content Distribution Networks (CDNs), where content objects are extensively replicated throughout the network closer to the user and the discovery (or resolution) could potentially become the bottleneck. Traditionally, the design of discovery schemes has assumed that all players have the same discovery performance requirements, thus resulting in homogeneous demand. In such a setting, the discovery schemes deliver a discovery service that is oblivious to the actual, possibly heterogeneous, discovery requirements - and valuations - of the different players. In reality however, the CNN site will likely value a higher discovery level more than a generic residential site. The first question posed in this work is therefore the following: should the design of discovery mechanisms account for discovery service differentiation? We attempt to answer this question by introducing the *multi-level discovery* framework which is concerned with the design of discovery schemes that can provide different service levels to different sets of players⁴.

Obviously, there is a cost associated with being discoverable. This could be the cost of distributing and maintaining information (state) about the identifiers to provide a certain discovery level. In the majority of current schemes, the discovery demand is insensitive to cost. Accounting for and sharing the cost of discovery is an interesting problem whose absence in current path discovery schemes has led to critical economic and scalability concerns. As an example, the Internet's BGP [11] control plane functionality is oblivious to cost. More clearly, a node (BGP speaker) that advertises a provider-independent prefix (identifier) does not pay for the cost of being discoverable. Such a cost may be large given that the prefix is maintained at every node in the DFZ (the rest of the network pays!). Such incentive mismatch in the current BGP workings is problematic and is further exacerbated by provider-independent addressing, multi-homing, and traffic engineering practices [12]. Notice here that BGP with its control and forwarding planes represents a discovery

This work is partially funded by the National Science Foundation (NSF) grant CNS-0626380.

¹The terms identifier, name, and virtual address are used interchangeably.

²The terms locator and label are used interchangeably in this context.

³This is "how easy" it is for the network to discover the player not the opposite.

⁴We note that Akamai [10] currently offers such expedited resolution service in a form that tightly couples content service providers with DNS providers.

scheme on prefixes which are technically flat identifiers in a largely de-aggregated namespace. Hereafter, we refer to this form of BGP as BGP-DA for De-Aggregation. Hence, *we conjecture that a discovery mechanism should be aware of valuations and cost necessitating that players/nodes pay for the distributed state they introduce into the system.* The distributed state is generally proportional to the discovery performance attained. A cost-sharing discovery mechanism determines “how discoverable” players are, and the payments or cost shares they have to make.

This paper presents a preliminary investigation of two main challenges pertaining to the design of identifier-based discovery mechanisms. The first challenge, which is the main focus of this paper, is that of differentiated discovery. Along this dimension, we present a proof-of-concept multi-level discovery architecture and we analyze its scalability properties. The second challenge is that of providing an economic model that accounts for cost and valuation in the design of discovery mechanisms. Along this dimension, we simply discuss the challenges inherent to such an economic framework and we present some open questions. By adding an economic dimension to the discovery design space, we hope to gain more knowledge about the complex design decisions pertaining to naming and discovery in networks, and to be able to design discovery mechanisms that are suitable for a future Internet. This paper reports on early results and lays out our research agenda for investigating the topic.

The rest of the paper is organized as follows: section II presents a taxonomy of discovery schemes based on their business models. The multi-level discovery problem is then introduced in section III and its feasibility is demonstrated in the context of compact routing. A general model for designing discovery mechanisms based on the Algorithmic Mechanism Design [13] framework is then overviewed in section IV before concluding, and briefly overviewing our main challenges, and our next steps in section V.

II. BACKGROUND

We start by modeling the network as a graph $G = (V, E)$ with a set of nodes V , $|V| = N$, where each node $u \in V$ can host at most a single default object ⁵. An object has a unique identifier. Hereafter, we shall refer to the objects by index $i = 1 \dots N$ and it should be clear that anytime we refer to node i or player i , we are actually referring to the default object hosted on the node. The objects are the players P demanding a level of discovery from the mechanism.

A. What is discovery?

Recall that an identifier represents an object’s identity and remains unchanged when location (e.g. topology) information changes. A locator identifies the location of an object and must change when the object’s location information changes. The notion of discovery throughout this paper refers to *path discovery based on identifiers*. A discovery scheme f_d (or

⁵The default object is meant to capture the case when the node itself is the object.

Fig. 1. Two layer model.

alternatively referred to as virtual routing) generally operates on top of a routing scheme f_p that routes based on locators. We refer to this model as the two-layer discovery model as depicted in Figure 1. Whenever f_p exists, all that remains to be discovered by f_d is the identifier-to-locator mapping (e.g. DNS name to IP address mapping). When f_p is not available, then path discovery needs to be performed by f_d as well (e.g. BGP-DA on provider-independent prefixes). The two-layer model may be applied recursively i.e. a new discovery function f'_d may operate on top of f_d where the latter is virtualized as the locator routing function. This paper is particularly concerned with the design of mechanisms that implement the discovery function f_d .

The main reason that we distinguish the two routing functions f_d and f_p is because there are instances where the two functions are managed by different entities that can minimally collaborate to jointly optimize the two functions. For example, with the current Internet where BGP implements some form of f_p (routing on IP addresses), discovery schemes are being introduced in a separate plane that is not necessarily provisioned by ISPs (players) themselves but rather by other economic entities (as in DNS, and recently [1]). On the other hand, in name-independent compact routing design [6], it is assumed that the two functions are being jointly optimized to achieve a single global goal of efficient communication/discovery. This requirement has motivated us to study discovery mechanisms separately and to deviate from the pure algorithmic treatment of the topic towards *solution concepts* that are based in economics.

B. Discovery models and schemes

Figure 2 shows some classic models used by current discovery schemes (and proposals) following the two-layer model. Big circles (light and dark) represent nodes used by f_p at the lower layer (nodes V). At the upper layer, big dark circles represent a subset of those nodes that maintains state about the virtual namespace (service nodes V_D where $V_D \subseteq V$); Small dark circles are the objects that wish to be discovered or the players (players P). Figure 2 tries to illuminate the relationship between the players P (who receive the discovery service), and the nodes V_D (who incur the discovery cost). This relationship is important in an economic setting, such as when studying pricing schemes and when devising a strategic model (and solution concept) for the problem at hand. For example, service nodes in model I (described shortly) are generally considered to be obedient (i.e. to follow the protocol) as they belong to the same administrative entity (or to multiple competing entities each providing the same service). In models

Fig. 2. Representation of some common models for discovery.

Model	Representative Schemes
model I	DNS, DONA [4], eFIT [3], LIS ([1], etc.)
model II	DHTs (Chord [15], etc.)
model III	NICR ([6], [7], etc.), BGP-DA, ROFL [5]

TABLE I
IDENTIFIER-BASED DISCOVERY SCHEMES.

II and III however one needs to consider strategic service nodes in addition to the strategic agents where the two sets could be the same (check [14] for more details). Some of the representative schemes in the literature that follow these service models are listed in Table I.

In model (I) [$V_D \neq P$], there is a dedicated set of nodes V_D (possibly infrastructure) that keep the state information about the virtual namespace while the players P reside on different nodes. DNS is one example of a centralized scheme that follows this model. In DNS, V_D is the set of root/gTLD servers and the players are domain servers that keep zone files. Another sample scheme that uses this model and that is distributed is the recent DONA proposal [4] where V_D is the set of resolution handlers, and the players are generally objects on edge nodes. Another set of proposals that fits under this model is embodied by the Locator-ID-Split (LIS) work which aim at providing discoverability to edge sites (e.g. [1]) or nodes (e.g. [2]) in the Internet.

In model (II) [$V_D = P$], the state is kept on the same set of nodes that the players reside on. In such a model, the players themselves have a common interest in implementing the discovery scheme f_d . The typical example here is Distributed Hash Tables (DHT).

In model (III) [$V_D = V = P$], the state is maintained on all the nodes V and the players are all the nodes. This model is common to proposals that perform native routing on flat identifiers. One class of schemes that fits under this model is represented by the Name Independent Compact Routing (NICR) [6]. In NICR, the upper and lower layer functions are jointly designed and closely related (more details on NICR later). Another class of schemes that belong to this model does not utilize an underlying f_p i.e. f_d is basically a simultaneous discovery and forwarding scheme. BGP-DA

is the representative scheme here where the players are the prefixes advertised by ASes V and where it is necessary for all nodes V to keep the state in order for prefix path discovery (i.e. routing in this case) to succeed. Another recent scheme is the DHT-based ROFL [5], in which the routers are the nodes (if we ignore objects here) that are themselves the players identified by flat identifiers (hashes).

It is worth noting that each of the schemes in Table I is designed to satisfy a set of requirements and is based on a set of assumptions about the two-layer functions. Some of the common requirements observed in the literature include *efficiency*, *scalability*, *user-control*, *robustness*, *trust*, *economic requirements*, etc. Some of assumptions address the underlying graph structure (e.g. *scale-free*, or *small-world*) assumptions, or more specific structural assumptions of underlying metric embeddings.

III. CAN WE DESIGN MULTI-LEVEL DISCOVERY (MLD) SCHEMES?

Before delving into the economic aspects of the mechanism (costs, valuations, and incentives), this section tries to answer the question of whether multi-level discovery is algorithmically feasible. We start by providing a generic definition of the MLD problem. The problem specifics will depend on the context, mainly the design assumptions and requirements.

Definition 1: Multi-level discovery (MLD) problem statement: Given a graph $G = (V, E)$, a set of nodes with unique identifiers (identifier of node i is simply $\langle i \rangle$), set of m discovery levels where each node is associated with some level $l \in \mathcal{L}$ ⁶, and possibly some underlying routing function f_p that routes on locators, devise a discovery scheme that routes on identifiers. The set $\mathcal{L}$ of possible discovery levels is known to all nodes. The scheme is expected to deliver to each node i in G its requested discovery level $l \in \mathcal{L}$.

The main challenges inherent to the MLD problem arise from the requirements that (1) different levels of service must be supported by the same scheme, and that (2) the discovery level of a destination $\langle t \rangle$ is unknown at the time of discovery.

A traditional class of discovery schemes that satisfies the single object per node assumption and that has been extensively investigated in the research community is the general Name Independent Compact Routing (NICR) problem first introduced in [6]. NICR is of particular interest to this section and we shall extend it for implementing a MLD scheme on trees. More specifically, we extend Laing's NICR scheme [16] which operates on top of the optimal Thorup-Zwick labeled routing scheme on trees [17]. The latter represents f_p over which the identifier-based discovery scheme is implemented.

A. NICR scheme on trees

A name-independent compact routing scheme on trees (NICRT) is developed by Laing [16] with a space/stretch tradeoff based on a parameter k . The scheme achieves stretch $2^k - 1$ for a space requirement of $\tilde{O}(k^2 n^{1/k})$, where n is the

⁶When the set of discovery levels is discrete, a level becomes a "class" of service.

number of nodes. From a high level perspective, the tradeoff is achieved by asking each node to know about a set Σ^i of nodes ($|\Sigma^i| = n^{i/k}$) at concentric circles or neighborhoods N^i , $0 \leq i \leq k-1$ from itself. Routing towards a destination d proceeds through prefix matching of d 's identifier $\langle d \rangle$ represented in base $n^{1/k}$ (denoted by $\langle d \rangle_{n^{1/k}}$). Delivery is guaranteed in at most k hops i.e. by matching the k letters of d 's identifier base $n^{1/k}$. The main idea is that as the value k increases (i.e. as the number of concentric circles or layers increases), a node will keep less information about the rest of the network but the stretch which is directly proportional to the number of layers will increase. On the other hand, as k decreases (i.e. fewer layers), a node will keep more information about the rest of the network and the stretch decreases accordingly.

Laing's scheme is based on a coloring theorem for trees. The coloring theorem states that any tree with n nodes can be colored with q colors such that every neighborhood $N_q(v)$ of size q (for every node $v \in V$) is distinctly colored i.e. each node in $N_q(v)$ has a unique distinct color from the set of colors $[q]$ (check [16]). The theorem is used in the NICRT scheme to uniquely color neighborhoods $N^i(v)$ of size $n^{i/k}$ at each layer $i, 0 \dots k-1$.

Laing's scheme works as follows: Given a tree $T = (V, E, w)$, and a $k \geq 1$, multiple layers of coloring are assigned to nodes as follows: at layer $1 \leq i \leq k-1$, T is fully colored with Σ^i colors where $|\Sigma^i| = n^{i/k}$ and $\Sigma = \{0, 1, \dots, n^{1/k} - 1\}$ is the alphabet. Note that the neighborhood of a node v is denoted by $N^i(v)$ and is the set of $n^{i/k}$ closest nodes to v including the latter. Hence $|\Sigma^i| = |N^i|$ and the coloring theorem achieves a full coloring. Each node $u \in V$ is hence assigned a unique color $c_i(u)$ at layer i , where $c_i(u) \in \Sigma^i$. In addition to the $k-1$ colors node u obtains, it has its unique identifier $\langle u \rangle$ picked from the set $\{0, \dots, n-1\}$ and represented in base $n^{1/k}$ and padded to the left with zeroes. Thus $|\langle u \rangle| = k$.

Storage: Each node u has an identifier $\langle u \rangle$ and $k-1$ colors $c_i(u)$. Denote by $\sigma_i(u)$ the length i prefix of $\langle u \rangle$. In addition to the labeled compact routing table information of [17]⁷, node u creates its routing table according to Algorithm 1.

Routing: In terms of routing to some destination t with identifier $\langle t \rangle$ starting at some source s , routing proceeds as indicated in Algorithm 2.

Note in Algorithm 2 that each next hop (i.e. v_{i+1}) is guaranteed to belong to $N^{i+1}(v_i)$. Note as well that the only node that matches $\sigma_k(t)$ is the node whose identifier is $\langle t \rangle$ which guarantees delivery [16].

Algorithm 1 Routing table construction for node u

```

1: for each layer  $i, 0 \dots k-1$  do
2:   Let  $\lambda = \{c_i(u), \sigma_i(u)\}$ , where  $c_0(u)$  and  $\sigma_0(u)$  are the empty string  $\epsilon$ 
3:   for each  $\tau \in \Sigma$  do
4:     store label of closest node  $v$  to  $u$  that satisfies  $c_{i+1}(v) = \lambda\tau$ 
       or  $\sigma_{i+1}(v) = \lambda\tau$ 
5:   end for
6: end for

```

⁷This information is used for optimal stretch-1 routing based on topological labels.

Algorithm 2 Routing to $\langle t \rangle$

```

1: let  $v_0 = s$ 
2: for each layer  $i, 0 \dots k-1$  do
3:   route to node  $v_{i+1}$  which is the closest node to  $v_i$  that matches
      $\sigma_{i+1}(t)$  i.e. node  $v_{i+1}$  satisfies  $c_{i+1}(v_{i+1}) = \sigma_{i+1}(t)$  or
      $\sigma_{i+1}(v_{i+1}) = \sigma_{i+1}(t)$ 
4: end for

```

B. Extending Laing scheme to support MLD

In the preceding scheme, the effect of the parameter k was to control the space/stretch tradeoff achieving stretch $2^k - 1$ for a space requirement of $\tilde{O}(k^2 n^{1/k})$. In this section, we extend Laing's scheme by allowing multiple stretch levels (or multiple values of k) on the same tree T for different sets of nodes. Discovery levels will correspond to values of k in Laing's scheme which directly determines the stretch.

More clearly, we assume the existence of a set $K = \{k_1, \dots, k_m\}$ ($m = |\mathcal{L}|$) of stretch levels ordered in ascending order with $\mathcal{L} \subseteq \mathbb{Z}^+$. Assume also without loss of generality that n is a k_m th power and that $k_1 \geq 2$. Each k_i corresponds to a discovery level $l = \frac{1}{2^{k_i}-1}$, $l \in [0, 1]$ and we assume that $m = |\mathcal{L}| = O(n^{\frac{1}{k_m}})$. Nodes in the mechanism design framework that we shall describe shortly in section IV will express valuations $v_i : \mathcal{L} \rightarrow \mathbb{R}$ over the set $\mathcal{L}$ of possible discovery levels. Having said that, the main idea that we shall use for extending Laing's algorithm to support multiple discovery levels on the same tree T introduces ACCELERATE tables that expedite discovery/routing for nodes that demand higher discovery levels. The extended scheme starts by providing the lowest discovery level ($\frac{1}{2^{k_m}-1}$) to all nodes by constructing Laing scheme for $k = k_m$. The pseudocode for construction of the routing tables is listed in Algorithm 3. Lines 6, 17 in Algorithm 3 and lines 5, 6 in Algorithm 4 encapsulate the main logic for expedited discovery.

In terms of routing to destination $\langle t \rangle$ using the extended scheme, we extend routing Algorithm 2 as depicted in Algorithm 4 given that each node knows the set of stretch levels $k_j, j = 1 \dots m$.

Analysis: It can be easily verified that delivery is guaranteed as well as $d(v_i, v_{i+1}) \leq 2^i d(s, t)$ in the extended algorithms (check [18]). In order to maintain the sub-linear space requirements at each node, the extra state maintained at each node for discovering higher level nodes must be less than a constant factor of $k^2 n^{\frac{1}{k}}$. First, at line 15 of Algorithm 3, in the worst case there are at most $n^{1-\frac{s}{k_m}}$ nodes in D_{k_j} that have the same length s prefix (when $|D_{k_j}| = n$) i.e. that can potentially introduce state on the same set of nodes B_s . Thus the maximum increase in any node's routing table size is $m \cdot n^{1-\frac{s}{k_m}}$. We have already assumed that the total number of levels $m = O(n^{\frac{1}{k_m}})$. Formally, in order to maintain sub-linear space at each node, the following condition must hold: $n^{1-\frac{s}{k_m}} \leq \alpha k_m^2 n^{\frac{1}{k_m}}$ for some large constant α , or $s \geq k_m(1 - \frac{\log \alpha k_m^2}{\log n}) - 1$. This constraint must hold when choosing the set of possible discovery levels $\mathcal{L}$ (and hence the respective set K) in order for the extended routing scheme to satisfy the sublinear space requirement inherent to compact routing design.

Algorithm 3 Extended table construction for node u

```
1: Let  $K' = \{k_1 - 2, \dots, k_{m-1} - 2\}$ 
2: for each layer  $i, 0 \dots k_m - 1$  do
3:   Let  $\lambda = \{c_i(u), \sigma_i(u)\}$ , where  $c_0(u)$  and  $\sigma_0(u)$  are the empty string  $\epsilon$ 
4:   for each  $\tau \in \Sigma$  do
5:     if  $i \in K'$  then
6:       store label of closest node  $v$  to  $u$  that satisfies  $c_{i+1}(v) = \lambda\tau$ 
7:       store label of closest node  $w$  to  $u$  that satisfies  $\sigma_{i+1}(w) = \lambda\tau$  only if  $w \in N^{i+1}(u)$ 
8:     else
9:       store label of closest node  $v$  to  $u$  that satisfies  $c_{i+1}(v) = \lambda\tau$  or  $\sigma_{i+1}(v) = \lambda\tau$ 
10:    end if
11:  end for
12: end for
    // Construct the ACCELERATE table
13: for each level  $k_j, j : m - 1$  downto 1 do
14:   Let  $s = k_j - 1$ 
15:   Let  $D_{k_j}$  be set of nodes requiring level  $k_j$ 
16:   for each node  $u \in D_{k_j}$  do
17:     Let  $B_s$  be set of nodes whose color at layer  $s$  is  $\sigma_s(u)$ 
18:     Add extra pointer  $\{<u>_{n^{1/k_m}} \rightarrow \text{label}(u)\}$  at each node  $v \in B_s$ 
19:   end for
20: end for
```

Algorithm 4 Routing to $<t>$ using extended scheme

```
1: let  $v_0 = s$ 
2: for each layer  $i, 0 \dots k_m - 1$  do
3:   if  $<t> \in \text{ACCELERATE table}$  then
4:     route directly to  $t$  using  $\text{label}(t)$ 
5:   else if  $i = k_j - 2$  for any  $j = 1 \dots m$  then
6:     route to node  $v_{i+1}$  which is the closest node to  $v_i$  such that  $c_{i+1}(v_{i+1}) = \sigma_{i+1}(t)$  or  $<v_{i+1}> = <t>$ 
7:   else
8:     route to node  $v_{i+1}$  which is the closest node to  $v_i$  that matches  $\sigma_{i+1}(t)$  i.e. node  $v_{i+1}$  satisfies  $c_{i+1}(v_{i+1}) = \sigma_{i+1}(t)$  or  $\sigma_{i+1}(v_{i+1}) = \sigma_{i+1}(t)$ 
9:   end if
10: end for
```

The major disadvantage inherent to most compact routing schemes is the fact that they are not concerned with the dynamics of the network and particularly with node churn. The schemes consider a static graph on which a data structure is constructed and do not worry about the construction algorithm and cost. However, our interest in compact routing in this section is primarily due to the mathematical bounds imposed on space and stretch which support our discovery level concept by providing guarantees on levels of performance. Having motivated the MLD problem and demonstrated its feasibility algorithmically, we now proceed to briefly introduce the mechanism design framework that accounts for costs, valuations, and incentives.

IV. GENERAL PROBLEM FORMULATION

The main goal of our work is to design identifier-based discovery mechanisms that are: (1) distributed (i.e. inputs and outputs of the mechanism are distributed throughout the network as defined in Distributed AMD [14]), (2) efficient (i.e.

the mechanism will try to maximize some concept of *social welfare*), (3) incentive-compatible (i.e. the players will not try to manipulate the outcome of the mechanism to their benefit), and most importantly (4) cost-sharing and possibly budget-balanced. Recall that budget-balance occurs when the global cost of the mechanism is offset by the payments made by the players. Note that the problem we are currently addressing assumes that all participating nodes cooperate to implement f_p and f_d in terms of routing and forwarding i.e. nodes do not try to computationally manipulate the routing/forwarding functions. The only strategic aspect of our current model is the players' valuations of discovery levels that are declared to the mechanism designer. Such assumption is directly applicable in schemes that follow model I in Figure 2 since the players can not manipulate routing when $V_D \neq P$.

A. The Discovery Mechanism

The ingredients of the discovery mechanism are: (1) an input valuation function $v_i(\cdot)$ for each player i where $v_i \in V_i$, the latter being the valuation function space $V_i \subseteq \mathbb{R}^{\mathcal{L}}$, (2) an output function $O : V_1 \times V_2 \dots \times V_N \rightarrow \mathcal{L}^N$ which utilizes some discovery scheme to deliver a discovery level profile $\mathbf{L}$ to the players, $\mathbf{L} \in \mathcal{L}^N$, and (3) a cost-sharing function $\xi : V_1 \times V_2 \dots \times V_N \rightarrow \mathbb{R}^N$ that distributes the payments $\mathbf{p}$ to the players, $\mathbf{p} \in \mathbb{R}^N$. Hence, the discovery mechanism $M : V_1 \times V_2 \dots \times V_N \rightarrow \mathcal{L}^N \times \mathbb{R}^N$ maps valuations to a discovery level profile and a payment profile. We shall briefly describe each of the ingredients next.

The valuation function: Each player i has a private valuation function $v_i : \mathcal{L} \rightarrow \mathbb{R}$, that assigns a real value to each possible discovery level $l \in \mathcal{L}$. Intuitively, a player will have a valuation that matches its true internal business requirements - user demand internal to the player/node will require a certain performance level to satisfy the demand.

The output function: An output of the mechanism is a decision that aggregates the players' valuations. More clearly, the output function $O : V_1 \times V_2 \times \dots \times V_N \rightarrow \mathcal{L}^N$ maps the players' valuations to a discovery level profile $\mathbf{L} \in \mathcal{L}^N$, delivered to the players. Denote by $(\mathbf{L})_i$ the element of vector $\mathbf{L}$ (i.e. the discovery level l) delivered to player i , and by $(\mathbf{L})_{-i}$ the profile delivered to the rest of the players. To deliver a discovery level profile, the mechanism relies on a discovery scheme, denoted as (D, Al) . The discovery scheme (D, Al) dictates 1) how the namespace registrations (or state) are distributed on the nodes $V_D \subseteq V$ (denoted by D), and 2) how the search queries are forwarded such that players will be discoverable (denoted by algorithm Al). Let S_u be the registration state maintained at node $u \in V_D$ and let S be the global state under D , i.e. $S = \bigcup_{u \in V_D} S_u$.

The cost-sharing scheme ξ : In addition to delivering a discovery profile, the mechanism implements a function ξ that distributes payments p_i to the players (objects) where p_i is the amount player i pays to the mechanism.

The cost function C : The cost function is defined by $C : \mathcal{L}^N \rightarrow \mathbb{R}^+ \cup \{0\}$ that assigns to each output profile a real cost for delivering the profile. Given that a scheme (D, Al) will assign a set of registrations S_u to each node u that delivers

an output profile $\mathbf{L}$, the total cost associated with $\mathbf{L}$ under some fixed scheme (D, Al) is $C_{(D, Al)}(\mathbf{L}) = \sum_{u \in V_D} cost(S_u)$ where $cost$ is the cost function of maintaining the S_u registrations at node u . In this sense, the cost we try to formulate is the control plane cost of the discovery scheme (D, Al) ⁸. The mechanism assumes the existence of some stable scheme (D, Al) and the cost is minimized over the argument S where the former could be suboptimal. By fixing the discovery scheme, the stability of the mechanism increases and the network complexity that might arise due to changes in v_i decreases.

Utility and welfare: The value that a player i obtains as a result of an output profile $(\mathbf{L})$ is simply her valuation of the discovery level she receives $v_i((\mathbf{L})_i)$ ⁹. The utility of player i is $u_i = v_i - p_i$. It is implicitly assumed here that the player's preferences are quasi-linear and that no externalities exist i.e. player only cares about the discovery level she receives and not about other player levels. The global welfare of all the players under a scheme (D, Al) is, $NW(\mathbf{L}) = \sum_i v_i(\mathbf{L}) - C_{(D, Al)}(\mathbf{L})$ which measures the total benefit obtained by all players independent of payments. A mechanism is said to be efficient if it maximizes the global welfare $NW(\mathbf{L})$ implementing a social choice function.

Our next steps aim at extending the model to investigate the general classes of mechanisms that are strategy-proof, efficient, and cost-sharing. Additionally, we will specialize the results to specific discovery models where discovery levels and cost structures can be quantified.

V. CONCLUSIONS, CHALLENGES, AND FUTURE WORK

The mechanism design model described above generalizes the purely algorithmic treatment of discovery schemes to account for economic factors and is a work in progress [19]. The framework allows for discovery service differentiation, a problem we refer to as multi-level discovery or MLD, while accounting for heterogeneous player valuations. We have demonstrated the feasibility of MLD schemes in the context of NICR.

While this work sets the stage for future investigation of the topic, some of the main challenges we are currently addressing include: (1) formulating a cost function that is tractable given its dependence on globally distributed state; (2) devising distributed implementations of the mechanism and studying their algorithmic complexity [14] for the different discovery models; and (3) investigating the feasibility of implementing the distributed schemes as scalable extensions to legacy discovery schemes such as BGP, DNS, and DHT as identified in [14]. Additionally, we are investigating other tools based in microeconomics that can better model incentives, cost structures, and demand in distributed discovery mechanisms. While it provides tractable results, algorithmic mechanism design [13] suffers from several limitations including (i) the simple one-shot model versus the more realistic repeated

dynamics that are prevalent in distributed settings, (ii) the obliviousness to malicious behavior, and (iii) the inability to concurrently account for demand and supply [20]. In this sense, part of our current agenda aims at designing practical incentive models for discovery in BGP that could simultaneously account for supply and demand.

REFERENCES

- [1] V. Fuller, D. Meyer, and D. Farinacci, "Lisp alternative topology (lisp+alt)," <http://tools.ietf.org/html/draft-fuller-lisp-alt-03.txt>.
- [2] R. Moskowitz, P. Nikander, and P. Jokela, "Host identity protocol architecture," RFC 4423, May 2006.
- [3] D. Massey, L. Wang, B. Zhang, and L. Zhang, "A scalable routing system design for future internet," in *ACM SIGCOMM workshop on IPv6 and the Future of the Internet*. New York, NY, USA: ACM Press, 2007.
- [4] T. Koponen, M. Chawla, B.-G. Chun, A. Ermolinskiy, K. H. Kim, S. Shenker, and I. Stoica, "A data-oriented network architecture," in *Proceedings of SIGCOMM'07*. Kyoto, Japan: ACM, August 27-31 2007.
- [5] M. Caesar, T. Condie, J. Kannan, K. Lakshminarayanan, and I. Stoica, "Roff: routing on flat labels," in *Proceedings of SIGCOMM 2006*. New York, NY, USA: ACM Press, 2006, pp. 363–374.
- [6] B. Awerbuch, A. Bar-Noy, N. Linial, and D. Peleg, "Compact distributed data structures for adaptive routing," in *ACM STOC '89*. New York, NY, USA: ACM, 1989, pp. 479–489.
- [7] I. Abraham, C. Gavoille, D. Malkhi, N. Nisan, and M. Thorup, "Compact name-independent routing with minimum stretch," in *ACM SPAA '04*. New York, NY, USA: ACM Press, 2004, pp. 20–24.
- [8] L. Bent and G. M. Voelker, "Whole page performance," Workshop on Web Content Caching and Distribution, Boulder CO, August 2002.
- [9] C. Huitema and S. Weerahandi, "Internet measurements: the rising tide and the dns snag," in *Proceedings of the 13th ITC Specialist Seminar on IP Traffic Measurement Modeling and Management*, ser. IPseminar. Monterey, CA, USA: ITC, September 18-20 2000.
- [10] "Akamai Technologies," <http://www.akamai.com>, 2008.
- [11] Y. Rekhter, T. Li, and S. Hares, "RFC 4271: A border gateway protocol 4 (bgp-4)," 2006.
- [12] D. Meyer, L. Zhang, and K. Fall, "Report from the iab workshop on routing and addressing," Internet RFC 4984, Sep 2007.
- [13] N. Nisan and A. Ronen, "Algorithmic mechanism design," in *Proceedings of the 31st ACM Symposium on Theory of Computing*, Atlanta, GA, May 1999.
- [14] J. Feigenbaum and S. Shenker, "Distributed algorithmic mechanism design: Recent results and future directions," in *In Proceedings of the 6th International Workshop on Discrete Algorithms and Methods for Mobile Computing and Communications*. ACM Press, 2002, pp. 1–13.
- [15] I. Stoica, R. Morris, D. Liben-Nowell, D. R. Karger, M. F. Kaashoek, F. Dabek, and H. Balakrishnan, "Chord: a scalable peer-to-peer lookup protocol for internet applications," *IEEE/ACM Trans. Netw.*, vol. 11, no. 1, pp. 17–32, 2003.
- [16] K. A. Laing, "Name-independent compact routing in trees," *Inf. Process. Lett.*, vol. 103, no. 2, pp. 57–60, 2007.
- [17] M. Thorup and U. Zwick, "Compact routing schemes," in *ACM SPAA '01*. New York, NY, USA: ACM, 2001, pp. 1–10.
- [18] K. A. Laing, "Name-independent compact routing in trees," Technical Report, Tufts University - [online] http://www.cs.tufts.edu/tech_reports/reports/2003-2/report.pdf.
- [19] J. Khoury, "Discovery in large-scale networks," Ph.D. dissertation, to appear 2009, university of New Mexico, work in progress.
- [20] S. J. Bauer and P. Faratin, "Assessing the assumptions underlying mechanism design for the internet," In *Proceedings of Economics of Networked Systems (NetEcon'06)*, Michigan University, Ann Arbor, US, 2006.

⁸Note that we are not accounting for the forwarding plane costs which could be handled through per query rewards. We are solely concerned with the initial cost of constructing and maintaining the state.

⁹This is the value of being globally discoverable or known at the expected level i.e. the value of being famous!.