Using Semantics to Learn About Routing Data for Improved Network Management in the Future Internet

John Strassner, Sung-Su Kim, and James Won-Ki Hong, *Members, IEEE*

***Abstract*—One of the most fundamental management aspects of the Future Internet is the representation of management and operational data. The vast majority of languages and data structures used by network device manufacturers, such as SNMP-based designs and Command Line Interfaces, are data-oriented and are incapable of representing semantics. In addition, such languages have no ability to represent business concepts, such as a Service Level Agreement, or higher-level concepts, such as the ability to maximize revenue for all users. This paper proposes a knowledge representation based on the combination of models, which represent facts, and ontologies, which enable machine-based reasoning and learning based on facts. We then define an approach to route information based on the underlying semantics of data. Our approach to semantic routing can be used for both semantic querying and network management tasks. It is implemented via a set of semantic overlay networks.**

***Index Terms*—information model, network management, ontology, overlay network, semantics**

I. Introduction

One of the most fundamental aspects of management is the representation of data. Currently, network management and operational data has been built by network device manufacturers to ease the management of their (vendor-specific) device or device families, and hence is inherently specific to that particular vendor. In addition, the vast majority of languages and data structures used by network device manufacturers, such as SNMP-based designs [1] and Command Line Interfaces [2], are data-oriented and are not conducive to representing semantic knowledge, such as the effects of a command, its hardware and/or software requirements, and the true meaning of what that command does. Furthermore, such languages have no ability to represent business concepts, such as a Service Level Agreement, or higher-level concepts, such as "maximize revenue".

This paper is part of our research in developing a new management approach for the future Internet that is backwards compatible with legacy implementations, yet able to take advantage of new clean-slate approaches. Specifically, this paper focuses on first, introducing a new approach to representing management data that is able to represent the semantics of data and second, creating a semantic routing overlay network that can be used to route information based on semantics, instead of an abstract identifier such as an IP address. For example, suppose we were interested in finding out the best way to drive from one location to another. Instead of having to know the set of specific IP addresses of the most appropriate sensors along several routes that could be taken, we could instead query for any traffic sensors that are located in a particular location, or near an intersection of generic point of interest, such as a museum. We could also query for other properties, such as weather or social events that were local to where we wanted to go, to determine if they might influence traffic. This example is difficult to handle, as (1) it is not based on keywords, and (2) we are searching for a range of different content in multiple nodes with the express aim of *combining* the content to form our answer, instead of retrieving an atomic answer.

Many network management tasks rely on finding data based on their semantics. Hence, our research seeks to use the same approach to manage the network as well as construct a generic semantic query and retrieval service.

The organization of this paper is as follows. Section 2 describes the difference between traditional and semantic routing. Section 3 defines our approach in detail, and Section 4 presents experiments to test our approach. Section 5 provides related work. Conclusions and proposed future work are contained in Section 6.

II. Traditional vs. Semantic Routing

This section compares traditional routing done in IP networks with our proposed semantic routing.

A. Traditional Routing

Routing algorithms for packet-switched networks are responsible for selecting a path in a network to send traffic [3]. The sending of traffic from a source to a set of intermediate nodes to a destination is called forwarding; hence, routing can

Manuscript received January 6, 2010. This work was supported in part by the WCU (World Class University) program through the National Research Foundation of Korea funded by the Ministry of Education, Science and Technology (R31-2008-000-10100-0).

John Strassner is with the Pohang University of Science and Technology (phone: +82 54 279 5605; fax: +82 54 279 5699; e-mail: johns@postech.ac.kr).

Sung-Su Kim is with the Pohang University of Science and Technology (email: kiss@postech.ac.kr).

James Won-Ki Hong is with the Pohang University of Science and Technology (email: jwkhong@postech.ac.kr).

be viewed as the process that controls forwarding. Most routing algorithms only use a single path; however, multipath routing can use multiple alternative paths. The advantage of multipath routing is that alternative paths can be leveraged to improve fault tolerance or security, or to increase bandwidth; its disadvantage is increased cost.

Most networks have little, if any, concept of semantics used in their routing. For example, it is common to compute the "best" route, where "best" is determined by some metric or set of metrics (e.g., time to forward) and not save any other routes that were found in the analysis. Examples of such algorithms include Dijkstra's algorithm [4], which is used in many protocols, such as Open Shortest Path First [5]. Various routing metrics [6] have been proposed to choose one path over another.

Different protocols can use different routing metrics and algorithms that are not compatible with each other, preventing them from being directly compared. This prevents selecting the best path from among multiple protocols that advertise the same route. Therefore, external heuristics are used to select among the different protocols by assigning each protocol a particular cost; then, the entry from the protocol with the lowest cost is used for each route. Typically, the routing table only stores the best route to a given node; additional information may be stored in other repositories specific to that protocol, but separate from the routing table, in order to control the size of the routing table. Finally, since routing metrics are specific to a particular routing protocol, routers that use multiple protocols use one or more heuristics to select between routes learned from different routing protocols.

B. *Our Semantic Routing Approach*

Conceptually, we want to develop a new set of routing metrics that are semantic in nature. This means that they can be used to represent non-functional factors, such as cost and availability, as well as user desires and needs. For example, with traditional routing, there is no way to say "give me the best connection no matter what the cost for this type of traffic, but give me the least expensive cost connection for this other type of traffic." In our approach, this is accommodated by using different semantic routing metrics for each traffic type.

Our approach can be viewed as a simplified form of multipath routing. First, a control decision is made in the beginning of the routing process that decides whether to use traditional or semantic routing. Note that this does not preclude the use of multipath routing in the traditional case. If semantic routing is used, then one or more semantic routes will be formed that try to connect the source to the destination based on the meaning of the information, as opposed to an address that has nothing to do with the meaning.

Our approach considers semantics as a primary metric in routing for those cases where semantic routing (as opposed to traditional routing) is used. Thus, we have different types of semantic metrics, just as traditional IP routing has different types of IP metrics. Examples of semantic metrics include high-level non-functional metrics, such as cost, availability, reliability, and quality, as well as lower-level metrics, such as how close the meaning of the search term is to the meaning of data in different repositories. In particular, this latter search can be a parallel search on multiple queries, something that is not possible with current Internet routing architectures. This is illustrated in Fig. 1, where different combinations of attributes in a single query can be used to select one or more terms in multiple databases.

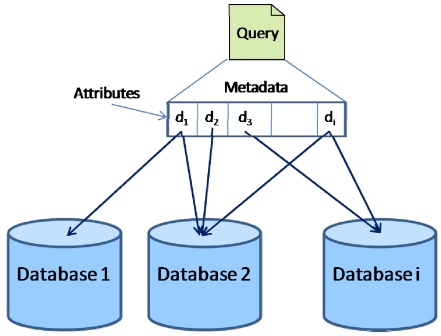

Fig. 1. Multiple Queries in Semantic Routing.

Our approach enables semantic routing to also use routing metrics, and hence be viewed as another protocol to be compared in a traditional routing process. More importantly, while semantic routing will use different algorithms compared to standard network routing, it is important to make semantic routing appear to be another routing process; otherwise, it will be very difficult to seamlessly integrate semantic routing with other routing processes. This would defeat the purpose of using semantic routing as an alternative to standard network routing.

III. Semantic Routing Design

We want to create an alternative routing model, one in which semantics can be used to manage both the creation of the overlay used to manage the network nodes as well as to efficiently search for and find knowledge and resources that are of interest to a particular query. However, we want to maintain backward compatibility. Therefore, instead of proposing a pure clean-slate design to replace IP routing, we instead propose an alternative approach that can use either IP routing or semantic routing, depending on the type of query and type of task being performed.

A. *Approach*

Our approach to making our semantic routing approach appear as another ordinary routing protocol is shown in Fig. 2. This enables semantic routing to be seamlessly combined with IP routing. In traditional routing, different interior routing protocols can each advertise the same route. There can be *m* different ways of routing a source to a destination, consisting of *i* semantic and *j* traditional routes. If two or more routing protocols (semantic and/or traditional) provide route information for the same destination, then we use a heuristic called administrative preference to select between them. However, since semantic routing is potentially more computationally complex than traditional IP routing, we only want to use semantic routing when it is needed. Therefore, we

define an additional heuristic, called semantic relevance, to weight the output of the semantic routing algorithms.

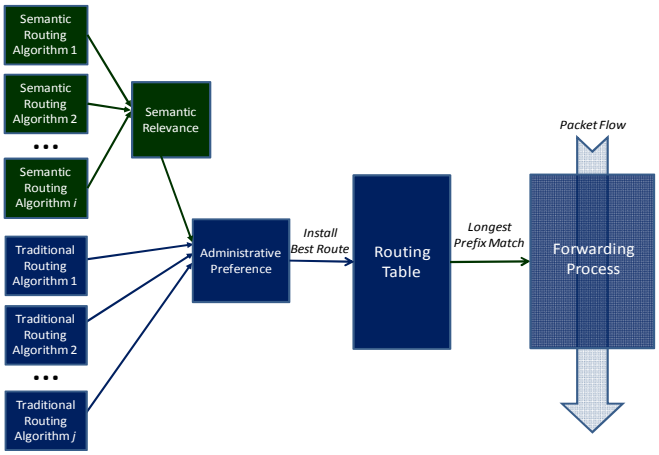

Fig. 2. Conceptual Overview of the Semantic Routing Process.

We can accommodate this by first, making the range of costs of the semantic route less than the costs in traditional routing and second, by multiplying the semantic routing weight by a semantic relevance factor, which is 1 if the routing is semantic in nature, and 10 if not. This has the effect of making the semantic route preferred if the request is semantic in nature, and most likely not preferred otherwise. The use of a separate semantic relevance parameter enables us to selectively adjust the use of semantic routing as we integrate it with traditional routing. We use the following weight ranges for each type of routing protocol:

- Most preferred interior routing protocol: 50
- Least preferred interior routing protocol: 200
- Most preferred semantic route: 7
- Least preferred semantic route: 50 (1)

B. Overlay Construction

We construct an overlay network on top of the physical network, where the overlay network provides a resource location service supporting application specific identifiers. The purpose of an overlay is for a group of peers $\mathbb{P}$ to provide access to a set of resources $\mathcal{R}$ by mapping $\mathbb{P}$ and $\mathcal{R}$ to an application-specific identifier space $\mathcal{I}$. A graph is embedded into the identifier space that enables any peer to access any resource in the network. The mapping enables different application services to be realized; these services, such as semantic routing, would in general be very hard to integrate into and support at the networking layer. These concepts are shown in Fig. 3. In a traditional IP network, the transport protocol operates on specialized addressing information contained in or associated with a message; the actual content associated with the message is typically not used. In contrast, a semantic network has no specialized addressing information; rather, it routes directly on the content of the message; this corresponds to "routing" in the overlay. The result of the overlay routing is then mapped to physical addresses in the underlying network, as depicted in Fig. 3.

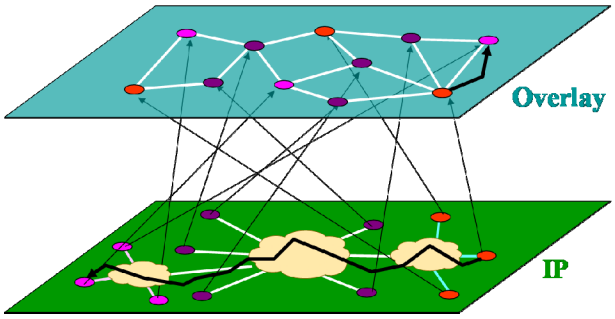

Fig. 3. Conceptual View of Semantic Overlay Network.

In traditional IP networks, messages are routed to a known destination. In our semantic network, we need to map a much larger address space than that used by IP. IPv4/v6 addresses have a limited range (0 to 2^{32} or 2^{128}). However, semantic addresses can have a much larger range, since the address space is the product of the number of all possible resources that have to be individually addressed and the number of possible ways to describe each resource. This latter factor is potentially very large, since we want to enable users to use multiple descriptions to address a resource (e.g., using synonyms or phrases). Therefore, we need a scalable network topology that can cover a large semantic routing space without having long routing times (i.e., long numbers of hops to reach a destination).

We satisfy the large address space problem by translating the user input into one or more common terms, and then searching on the terms. This is explained using Fig. 4.

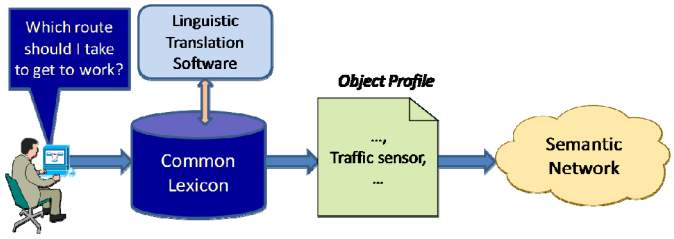

Fig. 4. Translating User Input into Attribtues using a Lexicon.

We define a common lexicon that contains the vocabulary used in the semantic network along with definitions, synonyms, and other lexical functions. The common lexicon enables us to *map* domain-specific vocabulary and concepts into a common form that is analyzed by our semantic network. For example, consider the earlier traffic sensor example. The semantic network must always contain knowledge to enable its results to be mapped to the underlying physical infrastructure; this knowledge is independent of the particular domains that are associated with the semantic network. The knowledge about traffic sensors is different than knowledge about other domains, such as weather data, even though we require data from both types of sensors in our example. Since the data describing traffic sensors is different than the data describing weather sensors, we store the data for each sensor type in its own knowledge base.

The lexicon contains software that translates the user's query into terms that are stored in the semantic network. This enables the user to use much of their native terminology, and not be limited by our system, while dramatically reducing the complexity and storage requirements of each node. This work is part of a larger linguistics-based effort that is beyond the scope of this paper.

In our system, we assume that each node can represent one or more data sources. We further assume that the data sources of one node all use the same local schema, but that in general, multiple schemata may be used in the system (e.g., the schema of one node may be different than that of another node). We solve this problem by introducing a common schema $\mathbb{S}$; this is constructed from the DEN-ng information model [7]. We choose this approach, rather than try to force the use of a common schema in each data source, since (1) there is no accepted common programming language today, and (2) device manufacturers have no business reason to retool their products to use such a language. Therefore, we build a set of mappings between the data model of each data source and our common schema $\mathbb{S}$. This is shown in Fig. 5.

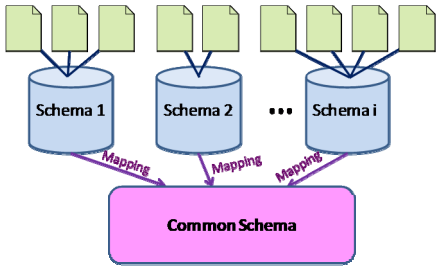

Fig. 5. Use of a Common Schema in our Semantic Overlay Network.

A particular document or object is characterized by a set of attributes that are defined in the common schema $\mathbb{S}$. These attributes can function in several different ways, including as a traditional keyword, a "see also" reference, or more complex semantics based on (for example) linguistic and/or ontological analysis. We define a *node profile* as the set of attributes $\mathbb{N}=\{\eta_1, \eta_2, \ldots, \eta_j\}$; this represents the set of attributes that a particular node is interested in. Hence, nodes will try and collect objects whose attributes match the attributes of the node's profile $\mathbb{N}$. The nodes express their interest in objects by registering their node profiles with the semantic network. Similarly, we define a client request, or query, profile for a object as a set of attributes $\mathbb{Q}=\{\rho_1, \rho_2, \ldots, \rho_j\}$. Queries are then matched by computing the dot product of $\mathbb{Q}$ and $\mathbb{N}$, as shown in Fig. 6:

$$sim(\mathbb{Q},\mathbb{N}) = \sum_{i=1}^{j} \rho_i \bullet \eta_i \qquad (2)$$

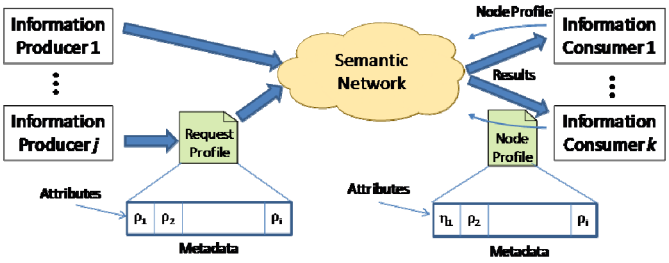

Fig. 6. Matching Information in the Semantic Network.

Note that node and request profiles can both be represented by the same set of attributes $\mathbb{A}$. The set of attributes $\mathbb{A}$ does not have to be large, as the set of keywords that are used in a system are much smaller than the number of objects that they refer to. For example, assume that a node is a traffic sensor, where each traffic sensor is located on an intersection in a city. A traffic sensor can be described in part by the attributes {traffic_sensor, street_name_1, street_name_2, city_name, zip_code}, where street_name_1 and street_name_2 are the names of the streets that intersect. In this example schema, the use of the traffic_sensor attribute enables a search to easily find all sensors that are used for traffic monitoring and not for other purposes; the search can then be customized by using one or more additional attributes to focus the search.

The node profile contains multi-dimensional information (i.e., the set of attributes $\{\eta_1, \eta_2, \ldots, \eta_i\}$) that identifies objects contained in that node. This means that we cannot use the notion of a static scalar address, since mapping a multi-dimensional structure to a single scalar unicast or multicast address results in the loss of information. Hence, we use a directed graph to represent our information. (We make our graph directed because, in general, communication is neither bi-directional nor equal in each direction between nodes.)

One way to represent this information is to use the W3C's Resource Description Framework (RDF) [8], which uses XML [9] as its syntax. Fig. 7 shows an example object profile for a traffic sensor, where a "tsnsr" namespace is defined, enabling the semantic attributes of traffic sensors to be defined.

```
<rdf:RDF
  xmlns:rdf = "http://www.w3.org/TR/WD-rdf-syntax#"
  xmlns:tsnsr = "http://schema.sensors.traffic/1.1">
  <rdf:Description>
    <tsnsr:Name>Traffic Sensor x453</tsnsr:Name>
    <tsnsr:GPS>
      <tsnsr:Position latitude="36°4'60N"
                      longitude="129°22'0E"/>
      <tsnsr:City>Pohang</tsnsr:City>
      <tsnsr:Province>Gyungbuk</tsnsr:Province>
    </tsnsr:GPS>
    <tsnsr:Streets>
      <tsnsr:Street1>Hyoja-Dong</Street1>
      <tsnsr:Street2><Daejam-Dong</Street2>
    </tsnsr:Streets>
...
   </rdf:Description>
</rdf:RDF>
```

Fig. 7. Example of an Object Profile for a Traffic Sensor.

Fig. 8 shows an example of a node profile that would be interested in the content of the traffic sensor object profile

shown in Fig. 7. In this example, the node profile defines a logical AND constraint, which means that all elements between it and its corresponding end tag must be true in order for the semantic network to match the object profile with the node profile. Two restrictions are defined that are part of this AND constraint; the first limits the location of the sensors to within two kilometers of a particular location, while the second looks for an exact match of a street named "Hyoja-Dong".

```
...
<node_profile:And>
  <tsnsr:Loc>
    <tsnsr:Position>
      <node_profile:distance within="2" units="km">
                            latitude="36"
                            longitude="129"/>
    </tsnsr:Position>
  <tsnsr:Loc>
  <tsnsr:Streets>
    <node_profile:string-match>Hyoja-Dong
    </node_profile:string-match>
  </tsnsr:Streets>
...
</node_profile:And>
...
```

Fig. 8. Example of a Node Profile that is Interested in a Traffic Sensor.

The object profile of Fig. 7 will match the node profile of Fig. 8 because the latitude and longitude values of the object profile are contained within the latitude and longitude value ranges of the node profile, and all other attributes are either the same (i.e., Street1) or do not care (e.g., since the object profile does not specify a name for the traffic sensor, it is defaulted to do not care). The matching can be easily done using, for example, a trie data structure [10] or one of its variants, such as a Patricia trie.

C. *Semantic Routing*

In our Semantic Network, routing is done very differently compared to routing in traditional IP networks. This is because (1) nodes are described by the meaning of content that they contain, instead of by artificial addresses that have no relationship to meaning or content, and (2) while IP addresses are scalar, semantics are multi-dimensional data. In our case, we use one or more entries from the set of attributes $\mathbb{A}$ to describe the meaning of data using node profiles to represent stored data. Routing is then done by taking the dot product of node profile and object profile attributes, as defined in equation (2).

There are many ways to design semantic routing; two of these are building a routing tree that is made up of a set of filters and turning the semantic network into a small-world network. Since our attributes are strings, a trie data structure is a natural choice, since searching for a key takes at worst case $O(\ell)$ time for a key of length ℓ, while binary search trees can take up to $O(\ell \log n)$ time, where n is the number of elements in the tree. More importantly, longest-prefix matching is simple to implement using tries.

If nodes are associated with a single object, the routing tree filtering approach works very well. However, if there are a large number of nodes, then it becomes increasingly difficult to guarantee a fast response, because there is no mechanism to keep the number of hops small. This is exacerbated when nodes are associated with multiple objects. In addition, when a node has multiple objects, there is no easy way to control the assignment of objects to nodes in an optimal manner.

This last problem is a clue to a novel solution. If we want to obtain an optimal semantic distribution of those objects (i.e., enable each object to be associated with a set of other objects that are semantically related to it), then we need to enable the topology to self-organize, based on the set of attributes $\mathbb{A}$ used to describe an object, such that an optimal match between $\mathbb{A}$ and $\mathbb{N}$ (the node's profile) occurs. The optimal match is computed using a semantic relatedness [11] measure defined by Jiang and Conrath [12], as it outperformed other measures for our specialized vocabulary. Semantic relatedness measures how close the meaning of one entity is to the meaning of another entity using one or more lexical relationships, such as synonymy (e.g., "bank" and "lending institution"), antonymy (e.g., "accept" and "reject"), meronymy (e.g., "court" is a part of "government"), and/or other domain-specific relationships.

Our Semantic Network is shown in Fig. 9.

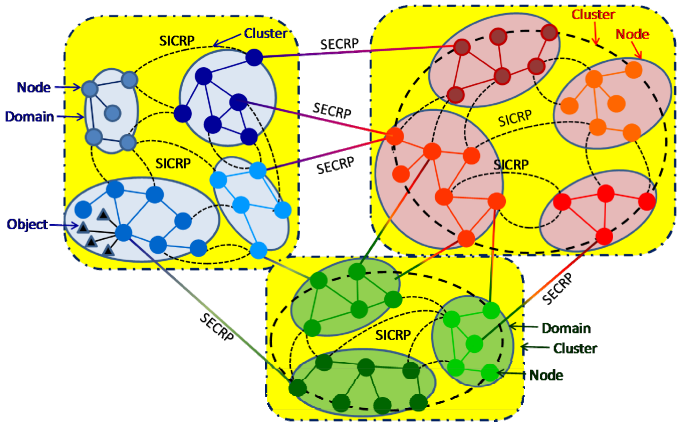

Fig. 9. Topology of our Semantic Overlay Network.

Speed in answering queries is directly proportional to the path length from source to destination. Instead of trying to build a single very large network, we build a large set of small networks, where each small network is a "cluster" of nodes that contain entities that are semantically related to each other. Each cluster is then connected into a small-word network [13] (described in Section D) to create the overlay. This enables us to rigidly control the shortest path in each of cluster, and then control the shortest overall path throughout the set of networks by making the set of networks a small-world network. Hence, we form our overlay network out of groups of clustered objects. This is similar in concept to the Internet, which consists of a large number of entities grouped into administrative domains called autonomous systems. Just as the Internet differentiates between intra- and inter-domain routing, we define two sets of protocols, a Semantic Exterior Cluster Routing Protocol (SECRP) for routing between clusters, and a Semantic Interior Cluster Routing Protocol (SICRP) for routing between nodes within a cluster. These two protocols are the subject of a future paper, and are beyond the scope of

this paper.

In our Semantic Network, each cluster consists of a set of domains; each domain contains a set of nodes; each node contains a set of objects. Nodes in different clusters communicate using the SECRP, while nodes in different domains within a cluster communicate using the SICRP. We define the following four sets of attributes for this topology to determine how to assign a given object to a particular node, how to assign a set of nodes to a domain, and how to assign a set of domains to a cluster:

- $\mathbb{A}_{\mathbb{o}}$ is the set of attributes $\{\alpha_1, \alpha_2, \ldots \alpha_i\}$ of a given object
- $\mathbb{A}_{\mathbb{n}}$ is the set of attributes $\{\alpha_1, \alpha_2, \ldots \alpha_j\}$ of a given node
- $\mathbb{A}_{\mathbb{d}}$ is the set of attributes $\{\alpha_1, \alpha_2, \ldots \alpha_j\}$ of a given domain
- $\mathbb{A}_{\mathbb{c}}$ is the set of attributes $\{\alpha_1, \alpha_2, \ldots \alpha_k\}$ of a given cluster

Each of the above three sets of attributes are subsets of the set of attributes $\mathbb{A}$ in our common schema $\mathbb{S}$, and each describes the set of semantics of that object, node, or cluster, respectively. Hence, we can group objects, nodes, domains, or clusters by the semantic relatedness of that type of entity. We use this to recursively construct our small-world network, where the attributes $\mathbb{A}_{\mathbb{o}}$, $\mathbb{A}_{\mathbb{n}}$, $\mathbb{A}_{\mathbb{d}}$, and $\mathbb{A}_{\mathbb{c}}$ are used to enable objects to group together in a node, nodes to group together in a domain, and domains to group together into a cluster, respectively. We can continue this hierarchy, such as grouping clusters into a larger cluster, to emulate administrative, geographical, physical, or other constraints, if necessary.

D. Using a Small-World Network for Semantic Routing

A small-world network has two attractive properties: (1) a low number of hops between any two randomly chosen nodes, which implies fast routing, and (2) a high clustering of nodes, which implies that a small-world network can quickly self-organize and provide good query capabilities, even under heavy demand. These properties are described by the clustering coefficient and the characteristic path length metrics in [14]. We change the formulation of these two metrics because we use a directed strongly connected graph, since in a network, routing between nodes can be asymmetric. Hence, we define a *weighted* version of the clustering coefficient by multiplying the contribution of the clustering of a node by how related its content is.

Watts et al. [13] proposed a β-model that models the topology of a small world network, as shown in Fig. 10. In this model, the topology is an interpolation of a regular network and a random network. That is, peers can be densely clustered, but have few characteristic paths. The β-model starts with an initial regular graph, and randomly replaces a lattice edge by a random edge with a probability β. The network is completely regular if β = 0, while it is completely random if β = 1. When β is set to an intermediate value between 0 and 1, the graph behaves like a small-world network.

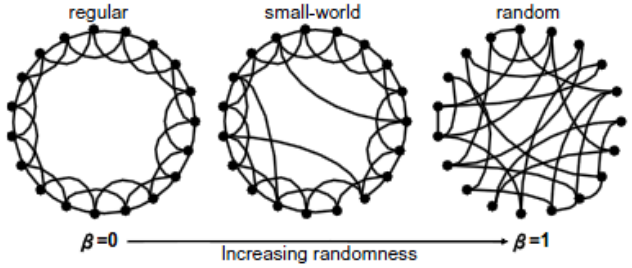

Fig. 10. Topology of our Semantic Overlay Network.

Conceptually, if the system starts with objects being randomly associated with different nodes, a network with small-world characteristics would enable the system to reorganize the nodes such that objects having similar attributes are grouped into similar nodes. For our system, this means that a node will advertise a node profile $\mathbb{N}$ consisting of a set of attributes $\mathbb{A}_{\mathbb{n}}$; this node profile is then used to match objects whose attributes $\mathbb{A}_{\mathbb{o}}$ are semantically related to the attributes of $\mathbb{A}_{\mathbb{n}}$. Similarly, nodes can then be grouped into domains, and domains can be grouped into clusters, by finding the highest semantic relatedness between $\mathbb{A}_{\mathbb{n}}$ and $\mathbb{A}_{\mathbb{d}}$ and $\mathbb{A}_{\mathbb{d}}$ and $\mathbb{A}_{\mathbb{c}}$, respectively. In every case, our "rewiring" procedure is based on computing the semantic relatedness between objects and nodes, between nodes and domains, and between domains and clusters, as follows.

We define four programmable thresholds, τ_{node}, $\tau_{inter\text{-}domain}$, $\tau_{intra\text{-}cluster}$, and $\tau_{inter\text{-}cluster}$, as follows:

$$\tau_{node} > \tau_{inter\text{-}domain} > \tau_{intra\text{-}cluster} > \tau_{inter\text{-}cluster} \quad (3)$$

Equation (3) uses the degree of semantic relatedness between two entities to organize our network as follows:

- τ_{node} attaches different objects to the same node
- $\tau_{intra\text{-}domain}$ attaches different nodes to the same domain
- $\tau_{inter\text{-}domain}$ attaches nodes in one domain in one cluster to nodes in a different domain in the same cluster
- $\tau_{inter\text{-}cluster}$ attaches nodes in one cluster in one domain to nodes in a different domain in a different cluster

Given the above thresholds, the network is organized as follows. An object is linked to a node if its semantic relatedness to that node is higher than τ_{node}. Note that we expressly avoid linking an object of one node to an object in a different node, as this would create too many links to manage. Rather, we "summarize" the contribution of a given set of objects to a node, and instead link nodes with high semantic relatedness. Once the objects are assigned to the nodes, the node profiles can optionally be recomputed to reflect the final distributions of objects that they contain.

A node is linked to a domain if its semantic relatedness to that domain is less than τ_{node} but greater than $\tau_{intra\text{-}domain}$. We again recompute the domain profile to reflect the final distribution of nodes that it contains. A node in one domain in a given cluster is linked to a different node in a different

domain in the same cluster if its semantic relatedness to the different node in that cluster is less than $\tau_{intra\text{-}domainr}$ but greater than $\tau_{inter\text{-}domain}$. A node in one domain in a given cluster is linked to a different node in a different domain in a different cluster if its semantic relatedness to that node is less than $\tau_{inter\text{-}domain}$ but greater than $\tau_{inter\text{-}cluster}$. If a node's semantic relatedness is less than $\tau_{inter\text{-}cluster}$, then a new domain in a new cluster is formed, since that node is not sufficiently semantically related to any other node.

Hence, the domain represents the "summarized" semantic relatedness of a set of nodes, and a cluster represents the "summarized" semantic relatedness of a set of domains. This enables the system to scale despite the increased computational complexity of computing semantic relatedness. In addition, by adjusting the four programmable thresholds, different views of the system can be formed.

Note that the protocols used for forming the links are different. The first two types of link (objects associated with a node in a given domain, and between nodes in the same domain), the *same* schema is used. Hence, no mapping is necessary to compute the semantic relatedness for these types of associations. However, for the last two types of links (inter-domain and inter-cluster), different protocols are used (SICRP and SECRP, respectively). This is because the schemata are different, and hence additional computations are required in order to map concepts and vocabulary between the different schemata.

E. Semantic Forwarding

In traditional IP routing, the destination address contained in the packet is looked up in the routing table of the router, and a longest prefix match is done; if a match is found, the packet is forwarded on the corresponding output port. If no match is found, then the packet is forwarded on the default link, which will drop the packet if the default link does not contain a default entry.

In our semantic network, the forwarding process consists of matching object and node profiles. If the two profiles match, then the message is forwarded on the ports that are associated with the matching node profiles. The matching operator can be configured to act as a direct match operator or as a filter. A default filter can be defined to catch messages that do not match any node, just as in the traditional case; this is useful for logging error messages for unmatched messages, or to check if a new cluster should be formed or not.

This becomes more complex if multiple objects are associated with a single node. For example, a node could be a home gateway serving multiple devices, or a node could be a database containing multiple documents. In this case, the node profile is an array of individual object profiles. In this case, the node profile can be represented in two different ways. One way is as a single bit vector, where a "1" means that one or more objects have the corresponding attribute, and a "0" means that none of the objects have the corresponding attribute. The advantage of this approach is simplicity, which translates to less storage. However, all this does is identify that a node has one or more objects that satisfy the request; additional work is still required to actually find the set of objects that match the query. The second way is as a j x k multi-dimensional array, consisting of k documents that each have j attributes. This enables the set of documents to be found in one step, but requires additional storage and processing power to find them. This tradeoff is quantified by the following experiment.

We defined four different types of sensor data in each node: 'traffic', 'temperature', 'wind', and 'humidity'. The first approach uses a vector to represent the type of data to be found, and then performs a search on the matched data; the second uses a j x k multi-dimensional array to find the requested document in a single operation. Fig. 11 shows the comparison between using j x k multidimensional array and using vector array.

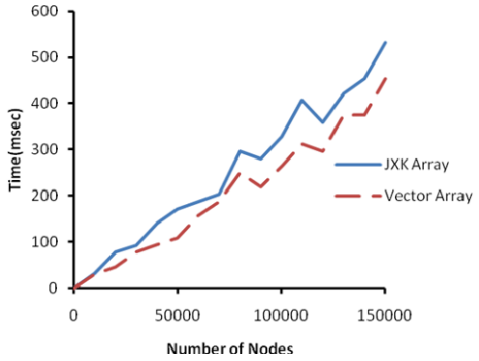

Fig. 11. Comparison of a Vector Array vs. a j x k Multi-dimensional Array

We assume that there are seven attributes to find data. If the number of matched attributes is greater than 4, the data is retrieved; otherwise, it is ignored. The time to find matched data is measured as a function of the number of nodes. As the results in Fig. 11 show, when the number of nodes are smaller than 10000, there is no difference between the two approaches. However, when the number of nodes exceed 10000, the time to find data using j x k multidimensional array is greater than the time required using vector array.

Fig. 12 shows the time to find sensor data that has various numbers of matched attributes using the j x k multi-dimensional and the vector array approaches. Fig. 12 shows that when the number of matched attributes is smaller than 4, the vector array approach yields shorter searching time compared with j x k multi-dimensional approach. This result means that if we want to find partially matched data, the vector array approach is more powerful.

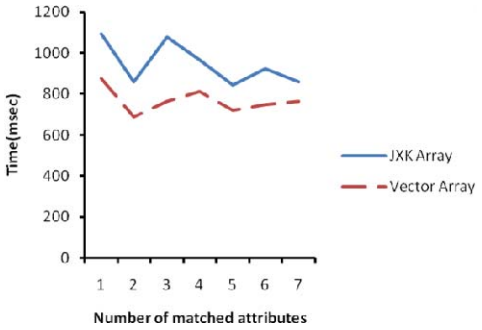

Fig. 12. Time to Find Data with different Number of Matched Data

IV. Related Work

There are a large number of P2P systems that operate in different ways. Pure P2P systems define their peers as having identical functionality. That is, every peer can retrieve files as well as supply them, meaning that searching is fully distributed. Examples of this type of system include Gnutella, Freenet, and Limewire [15]. These systems exhibit strong resilience; any node can join or leave the network without adversely affecting the ability to find content. However, the main drawback is lack of scalability.

Hybrid P2P systems define two types of nodes: regular nodes and a set of stand-alone nodes that are used specifically to support search functionality. The regular nodes use metadata to describe their contents, which is then indexed by the special nodes. A regular node contacts the special nodes that index the contents of the system to find appropriate content; the search nodes are used to find content in the regular nodes and direct the request to an appropriate node. Examples of this approach include Napster and Pointera [8]. Advantages include: (1) searching can be more efficiently performed in a centralized manner, and (2) metadata can enhance the searching facility. However, this is also a drawback, because it is impossible to predict which set of keywords will be used, and hence, different keyword-based searches can provide different results. In addition, the centralized functions used in these systems cannot scale.

Super-peer based systems, such as Edutella [9], contain two types of nodes: super-peers that index content and regular nodes that query for content. A super-peer acts as an indexing server to a set of regular peers (like the hybrid system), called a cluster. Each super-peer indexes the content of the peers connected to it. However, super-peers are also connected to each other (as in a pure system) and collaborate by submitting and answering queries on behalf of regular nodes and themselves. This enables a super-peer to forward the query to other super-peers if it cannot satisfy the query. The drawback is the additional overhead required for maintaining the super-peer connections, along with the extra traffic generated and processes necessary to keep the indices aligned.

Our semantic overlay network functions in a manner similar to a P2P network. However, we use the concept of semantic relatedness to organize content, and construct our network to exhibit small-world techniques.

V. Conclusions

Progress in the management of the future Internet and networked applications will increasingly depend on being able to understand the semantics of monitored data. This work notes the similarity between this task and previous work in social networks, where the desire to find data based on meaning, as opposed to addresses, plays a significant role. We seek to design a system that can support semantic routing for both of these important use cases.

We have designed a semantic routing system that can seamlessly integrate with existing approaches. It is implemented as an overlay, avoiding the physical disruption of the network. It appears as another interior routing process, facilitating its incorporation into existing network topology designs. This is in spite of its significant differences in routing and forwarding compared to traditional IP networks.

This first paper outlines our approach and shows the viability of attribute-based search and retrieval. Future work will expand on this foundation, and compare this approach to pure DHT systems and conduct more extensive tests.

References

[1] D. Harrington, R. Preshun, and B. Wijnen, “An Architecture for Describing Simple Network Management Protocol Management Frameworks”, RFC3411, STD0062, December 2002

[2] Cisco, http://www.cisco.com/warp/cpropub/45/tutorial.htm

[3] D. Medhi, K. Ramasamy, “Network Routing: Algorithms, Protocols, and Architectures”, Morgan Kaufman, San Francisco (2007)

[4] E. Dijkstra, “A Note on Two Problems in Connexion with Graphs”, Numerische Mathematik 1: 269-271

[5] J. Moy, “OSPF Version 2”, RFC 2328

[6] R. Baumann, S. Heimlicher, M. Strasser, A. Weibel, “A Survey on Routing Metrics”, TIK Report 262, ETH-Zentrum, Switzerland, 2007

[7] J. Strassner, “Introduction to DEN-ng”, Tutorial for FP7 PanLab II Project, January 21, 2009, available from: http://www.autonomic-communication.org/teaching/ais/slides/0809/Introduction_to_DEN-ng_for_PII.pdf

[8] http://www.w3.org/TR/REC-rdf-syntax/

[9] http://www.w3.org/TR/xmlschema-0/

[10] E. Fredkin, “Trie memory”, Communications of the ACM, Vol 3, Number 9, pages 490-499, 1960

[11] E. Gabrilovich, S. Markovich, “Computing Semantic Relatedness using Wikipedia-based Explicit Semantic Analysis”, Proc. Of the 20th Intl. Joint Conference on Artificial Intelligence, pages 1606-1611, 2007

[12] J. Jiang, D. Conrath, “Semantic similarity based on corpus statistics and lexical taxonomy”, Proc. of the Intl Conference on Research in Computational Linguistics, pages 19–33, 1997

[13] D. J. Watts, “Networks, Dynamics, and the Small-World Phenomenon”, American Journal of Sociology, Vol. 105, Number 2, pages 493-527, Sept 1999

[14] D.J. Watts, S. Strogatz: “Collective dynamics of ’small-world’ networks”, Nature 393 (1998), pages 440–442

[15] http://www.gnutellaforums.com/

[16] B. Yang, H. Garcia-Molina, “Comparing Hybrid Peer-to-Peer Systems”, Proc. of the 27th Int. Conference on Very Large Data Bases (VLDB 2001), pages 561–570, Italy, 2001

[17] W. Nejdl,W. Siberski, M.Wolpers, C. Schmitz, “Routing and clustering in schema-based super peer networks”, 2nd International Workshop on Peer-to-Peer Systems, 2003