MeGaDiP: a Wide-Area Media Gateway Discovery Protocol [1]

Dongyan Xu, Klara Nahrstedt, Duangdao Wichadakul
Department of Computer Science
University of Illinois at Urbana-Champaign
{d-xu, klara, wichadak}@cs.uiuc.edu

Abstract

In wide-area and heterogeneous environments, it is necessary to deploy media gateways at strategic locations within the networks, in order to deliver customized and composable multimedia services to individual users. In many cases, the media gateways have to be dynamically discovered, because (1) a host may not know a priori all the media gateways it will possibly use, and (2) the change of host location or end-to-end resource condition can make a known media gateway no longer valid. However, current general service location mechanisms are not sufficient to perform media gateway discovery. In this paper, we present MeGaDiP, a wide-area Media Gateway Discovery Protocol. It is based on the same basic architecture of the general service location mechanisms, and can be seen as a heuristics to be used first when discovering a media gateway. We also propose an extension of MeGaDiP using a hierarchical architecture to further improve the discovery success rate. The initial performance results from both our prototype implementation and simulation show the soundness of MeGaDiP.

I. INTRODUCTION

To provide multimedia services in heterogeneous environments, it is necessary to deploy media gateways at strategic locations within the networks. A media gateway intercepts a multimedia stream from a source host, performs certain processing on the media data, and forwards the processed media stream to the destination host. There are many reasons for using a media gateway, such as media scaling, format transformation, error control, caching, and prefetching. In many cases, the media gateways have to be dynamically located, because (1) a host may not know a priori all the media gateways it will possibly use, and (2) the change of host location or end-to-end resource condition can make a known media gateway no longer useful.

General service location mechanisms have been proposed[1, 2]. However, the services they consider are primarily 'request-reply' or 'sink-like' services (for the rest of this paper, we will call them 'RR/SL services'). The media gateway discovery problem is different from the RR/SL service location: the media gateway discovery is constrained by two end hosts - the source and the destination , instead of by one client in the RR/SL service. In addition, current service location mechanisms do not emphasize the performance of the service provider to be located. However, for a multimedia application, the end-to-end QoS is very sensitive to the service quality of the media gateway.

In this paper, we present MeGaDiP, a wide-area Media Gateway Discovery Protocol. Based on the same basic architecture, MeGaDiP is an easy extension of the general service location mechanisms. In fact, MeGaDiP can be seen as a heuristics to be used first for the discovery of a media gateway. The key properties of MeGaDiP are the following: (1) it is aware of *both* end hosts - any discovered media gateway will *not* create an unnecessarily long path between the end hosts *via the gateway*; (2) it is resource-aware - any discovered media gateway is likely to have sufficient end-to-end resources to perform its service; (3) it returns discovery results with low latency and high validity by caching and resource validation; (4) it only introduces small management traffic. Furthermore, an extension of MeGaDiP using a hierarchical architecture is proposed to improve the discovery success rate.

We are currently implementing MeGaDiP as part of the 2K System[3] - a component-based distributed operating system which supports flexible configuration and adaptive execution of distributed multimedia services. In 2K, multimedia services are *dynamically composed* by choosing the appropriate service components to fit the end-to-end resource conditions for individual clients. The role of MeGaDiP is to dynamically discover intermediate service components on media gateways. The rest of the paper is organized as follows: Section II describes the basic architecture of the general service location mechanisms, and discusses their inadequacy in media gateway discovery. Section III presents the details of MeGaDiP, and Section IV suggests a further extension of MeGaDiP. Section V presents our experimental results. Section VI discusses related work. Finally, we conclude this paper in Section VII.

[1]This work was supported by the National Science Foundation under contract number 9870736, the Air Force Grant under contract number F30602-97-2-0121, and National Science Foundation Career Grant under contract number NSF CCR 96-23867.

II. General Service Location

A. Basic Architecture

The entities in the basic architecture of general service location mechanisms include the *end hosts (EHs)*, the *service providers*, and the *dealers*, as shown in Figure 1[2].

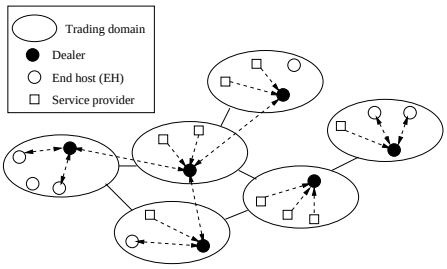

Figure 1: Entities in the basic service location architecture

- End hosts (corresponding to the *user agents* in the Service Location Protocol (SLP)[1]): an EH is the requester of a service. In order to find the provider of a desired service, the EH will query its local dealer (defined below) for any qualified service provider.
- Service providers (corresponding to the *service agents* in SLP): a service provider performs and delivers a certain service. In order to advertise its service, the service provider sends *service advertisement* to its local dealer. The service advertisement is sent periodically, so that the dealer can maintain the (soft) state of the service provider.
- Dealers (corresponding to the *directory agents* in SLP): The dealers are the key entities in the architecture. As an agent between EHs and service providers, the dealer accepts service advertisements from the service providers, and returns information about qualified service providers to requesting EHs. In the wide area environment, each dealer is associated with a *trading domain*. Hosts (including EHs and service providers) within the trading domain refer to the dealer as their *local dealer*. A service provider sends service advertisements only to its local dealer, and an EH only queries its local dealer. In order to support wide area service location, a dealer may have to contact other dealers for the forwarding of service queries, or for the return of query results found in its trading domain.

Based on the basic architecture, we now generalize two typical approaches used in current service location mechanisms:

- In the *push-based* approach, the dealer further pushes the service advertisements from its local service providers to other dealers. The service advertisement push is typically by periodic unicast or multicast. As a result, each dealer will gradually collect service advertisements of service providers outside its own trading domain. When a local EH submits a service query to the dealer, the dealer looks up the service advertisement entries for both the local and non-local service providers in order to find the qualified service provider.
- In the *pull-based* approach, when an EH queries its local dealer, if the dealer cannot find a qualified service provider from its local service advertisements, it will call for (pull) service advertisement(s) satisfying the query condition from other dealers. When a dealer with a satisfying service advertisement receives such a query, it will return the service advertisement to the requesting dealer. In order to save the pull bandwidth, the dealer usually uses *increasing-scope multicasts* to gradually expand the searching space. Furthermore, to speed up the discovery process, each dealer will cache the service advertisements from other dealers for future use.

B. Problems with Media Gateway Discovery

The current service location mechanisms work well for the location of RR/SL service providers (such as a web server or a printing server). However, there are problems with the discovery of a special class of service providers: the media gateways. In the discovery for a RR/SL service provider, the searching space can be roughly thought of as a *circle* of trading domains - the center of the circle is the local dealer of the requesting EH. However, in media gateway discovery, there are two EHs (the source and the destination) involved. Due to constraint of the end-to-end path between the two EHs, the searching space is intuitively more like a *stripe* of trading domains, starting from the trading domain of one EH, and ending in the trading domain of the other EH. Consequently, if a general service location mechanism is used for media gateway discovery, the following problems will arise:

(1) It is unaware of the end-to-end path between the two EHs. More specifically, it does not check if the gateway deviates significantly from the path between the source and the destination EHs. For example, in Figure 2, to find a media gateway between a source EH in Denver, Colorado and a destination EH in New York City, it is possible that the search result will be a media gateway G in Phoenix, Arizona, although there exists a better choice of media gateway G' in Chicago. The former choice is likely to introduce higher end-to-end delay (both in number of hops and in real time) than

[2]For notation consistency with MeGaDiP, the names of the entities in this architecture may not be the same as those in the literatures on the general service location mechanisms.

the latter.

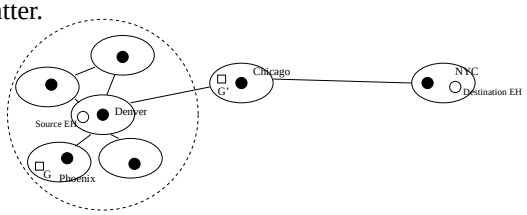

Figure 2: The unnecessarily long path problem

(2) It is unaware of the intermediate network condition between the two EHs. The media gateway discovery requires that there be sufficient network bandwidth from the source EH to the media gateway, and from the media gateway to the destination EH. In Figure 3, for example, the local dealer of EH s tries to discover a media gateway with MPEG-to-H.261 transcoding service, which can lower the data rate of a video stream before crossing the network bottleneck link l. However. it is possible that the search result is gateway G, which is of little use because G is 'behind' the bottleneck link l with respect to the video stream.

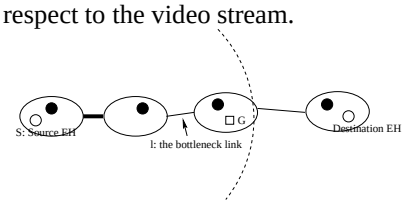

Figure 3: The bottleneck unawareness problem

(3) It may generate a large number of search results. For example, if a gateway G is found, it is likely that many other gateways within the same circle of searching space will also be returned as search results. With the expansion of searching space, the number of potential search results may increase rapidly. This will cause confusion at the local dealer, who does not know which media gateway is the most appropriate one to return to the requesting EH.

(4) Finally, it may incur excessive push or pull bandwidth. The local dealer discovers the qualified gateway(s) outside its trading domain by either the push or pull-based approach. As indicated in (3), the large but unnecessary number of search results will incur excessive push or pull bandwidth.

III. MeGaDiP: a Media Gateway Discovery Protocol

To remedy the inadequacy of the general service location mechanisms, we propose MeGaDiP, a wide-area Media Gateway Discovery Protocol, based on the basic architecture described in Section II.

The basic idea of MeGaDiP is the following: the discovery procedure starts from the local dealer of one of the EHs, and the seaching space is the stripe of trading domains along the end-to-end path between the source and the destination EHs, as shown in Figure 4. A dealer in this searching space will forward the query for the media gateway to the next dealer, if it can not find a qualified media gateway in its own trading domain. Finally, if no qualified media gateway is found by MeGaDiP, the system can fall back to the general service location mechanisms. The idea of giving discovery priority to the trading domains along the path between the EHs is based on the heuristics that a media gateway on or close to the end-to-end path is likely to find more bandwidth and/or to incur smaller end-to-end delay. This heuristics is well supported if QoS routing is the underlying routing mechanism; even for today's Internet routing, in most cases, the actual path between the EHs are not drastically different from the optimal route with the highest bandwidth and the lowest latency. Key issues of MeGaDiP are discussed in the following subsections.

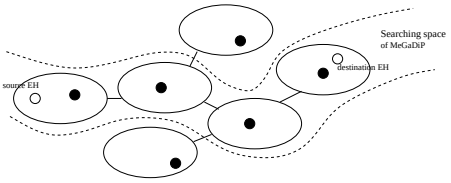

Figure 4: The searching space of MeGaDiP

A. Query Forwarding Order

The first issue of MeGaDiP is the order in which the stripe of along-the-path trading domains are searched. More specifically, let s and d be the source and destination EHs, and D_s and D_d be their local dealers, respectively. The question is whether the discovery procedure should start from D_s, and the query be forwarded towards D_d, or vice versa. In MeGaDiP, this is determined by the nature of the media gateways to be found. We categorize media gateways into two classes, depending on the relation between the media gateway's input data rate R_{in} and the output data rate R_{out}.

- Class I - $R_{in} > R_{out}$: a Class I media gateway processes an input media stream, and the corresponding output stream has a lower data rate. For example, the media gateway for the transcoding of Motion JPEG video to low-bit-rate H.261 video belongs to class I. A class I media gateway should be located closer to the source EH than to the destination EH, in order to save the network traffic of higher data rate (R_{in}) from s to the media gateway. Therefore, to discover a class I media gateway, the discovery will start from D_s;
- Class II - $R_{in} \leq R_{out}$: a class II gateway accepts a media stream at a data rate lower than the corresponding output stream. For example, the video

prefetching gateway belongs to Class II. A class II media gateway should be located closer to the destination EH than to the source EH, in order to save the network traffic of higher data rate (R_{out}) from the media gateway to d. Therefore, to discover a class II media gateway, the discovery will begin from D_d.

B. Along-the-Path Trading Domains

The next issue is to determine the trading domains along the path between the EHs. We make use of the Domain Name System (DNS) and the network routing mechanism to determine the trading domains. In MeGaDiP, to find the local dealer of any host, a DNS lookup is performed. We propose a new *Resource Record type* 'LD' in the DNS, which represents the address of the host's local dealer. More specifically, the DNS now provides a function *DNS_LD*: given a host name h, the corresponding local dealer of h is:

$$D_h = DNS\_LD(h) \quad (1)$$

This new resource record type can be deployed incrementally by the DNS servers over the networks.

For the two EHs s and d, each of them performs a DNS lookup to find its local dealer (D_s or D_d). To determine the dealers of other along-the-path trading domains, D_s first obtains a list of the intermediate routers r_i on the path from D_s to D_d. This is done by calling a simplified version of the *traceroute* routine. With the list of routers, the list of intermediate dealers (D_i) between D_s and D_d can be obtained by performing DNS lookups using the LD Resource Record type, i.e. $D_i = DNS\_LD(r_i)$[3].

The overhead of performing the simplified traceroute and performing the DNS lookups are both non-trivial. To improve efficiency, we propose that each dealer keeps the addresses of its immediate neighbor dealers, and caches the source and destination dealers of the paths that frequently travel across this trading domain.

C. Resource Awareness

One of the key properties of MeGaDiP is its resource awareness. The resources include the intermediate network bandwidth and the local resources at the media gateway.

First, MeGaDiP should ensure that there is sufficient bandwidth between the source EH and the media gateway discovered, and between this media gateway and the destination EH. More specifically, to avoid the bottleneck unawareness problem shown in Figure 3, a dealer should not always forward a query to its neighbor dealer when it can not find the media gateway in its own trading domain. In MeGaDiP, to be aware of the network condition in its neighborhood, each dealer periodically measures the bandwidth between itself and its neighbor dealers (in both directions)[4]. The dealer will use the measured bandwidth to decide if it should forward a query to its neighbor dealer.

Second, to be aware of the local resources of the media gateways, each media gateway in MeGaDiP reports its current resource availability in its periodic service advertisement to the local dealer. A media gateway's local resources include its (1) input bandwidth,(2) output bandwidth, (3) other critical resources (different media gateways have different critical resources, for example, CPU for a video transcoding gateway, and memory and disk for a video prefetching gateway).

D. Basic Discovery Procedure

We are now ready to describe the basic discovery procedure of MeGaDiP as follows.

(1) To discover a class I (II) media gateway, the source (destination) EH submits a query to its local dealer D_s (D_d). The condition $Cond$ of the query includes the following: (a) the name of the service provided by the media gateway, (b) the estimated input bandwidth R_{in} and output bandwidth R_{out} needed by the media gateway, and (c) the estimated amount(s) of critical local resource(s) needed by the media gateway. The query is in the form of $< D_s, D_d, Cond, gateway\_class >$.

(2.1) The dealer searches its service advertisement entries. If there exists a service advertisement that satisfies $cond$, the qualified media gateway is discovered and the result is returned to the requesting EH by the originating dealer. The discovery procedure terminates with a success.

(2.2) If there is no qualified media gateway in its trading domain, then the dealer finds the next dealer D_{nxt} to forward the query, as described in Section B.

(2.2.1) For a class I media gateway, D_{nxt} is the current dealer's *downstream* neighbor dealer (with respect to the media stream). If the current dealer is already D_d, (i.e. D_{nxt} does not exist), or the measured bandwidth *from the current dealer to* D_{nxt} is less than R_{in}, the current dealer reports failure to the originating dealer D_s. Otherwise, the query is forwarded to D_{nxt}, which will execute from step (2.1).

(2.2.2) Symmetrically, for a class II media gateway, D_{nxt} is the current dealer's *upstream* neighbor dealer (with respect to the media stream). If the current dealer is already D_s (i.e. D does not exist), or the measured bandwidth *from* D_{nxt} *to the current dealer* is less than R_{out}, the current dealer reports failure to the originating dealer D_d. Otherwise, the query is

[3]There may be duplications or non-defined values in $D_1, D_2, D_3, ...$, which should be discarded.

[4]MeGaDiP does not specify how the bandwidth is measured. In fact, there exist many tools or services for this purpose, such as Pathchar[4], Remos[5] etc.

forwarded to D_{nxt}, which will execute from step (2.1).

E. Discovery Results Caching and Validation

To speed up the basic discovery procedure, we propose the technique of caching and validation of discovery results in MeGaDiP. After a media gateway is discovered, its service advertisement is returned to the originating dealer. The originating dealer will cache the returned service advertisement for future use. Each cached service advertisement will be tagged with (1) the local dealer of this media gateway, and (2) the local dealer of the destination (source) EH (for a class I (II) gateway) involved in the discovery. When the originating dealer receives another query about the same type of media gateway, *and* the destination (source) EH (for a class I (II) gateway) has the same local dealer as tagged in (2), the originating dealer will return this service advertisement to the querying EH without starting the discovery procedure. The caching of discovery results significantly reduces the latency of MeGaDiP. The replacement algorithm for the cache can be the simple LRU algorithm.

However, a cached discovery result needs to be validated/invalidated regarding the end-to-end resource availability, otherwise the corresponding media gateway may not be able to provide satisfactory service quality to the requesting EHs. To validate the local resource availability on the media gateway, the dealer with the cached result will periodically contact the local dealer of the media gateway[5], obtain its current local resource availability information, and update (or delete - when the resources are run out) the cached service advertisement.

To validate the available end-to-end bandwidth from the source EH to the media gateway, and from the media gateway to the destination EH, we propose a simple and scalable solution which does not introduce any additional measurement traffic. In MeGaDiP, we use the media streams themselves as the measurement traffic. During a media streaming session, if the media gateway detects either of the following conditions[6]:

- Its actual input data rate R_{in}^{act} is less than $min(R_{in}, R^{src})$ for more than T amount of time (R_{in} is the input data rate specified in the media gateway query, R^{src} is the data sending rate at the source EH, and T is a predefined value);
- The data receiving rate at the destination EH R^{dst} is less than $min(R_{out}, R_{out}^{act})$ for more than T amount of time (R_{out} is the output data rate specified in the media gateway query, R_{out}^{act} is the actual output data rate of the media gateway).

the media gateway will send an *invalidation message* to the local dealer of the source (destination) EH, if this is a class I (II) gateway. The dealer, who has a cached service advertisement of this media gateway, will then invalidate and delete the advertisement, and start the discovery procedure to find a *different* qualified media gateway.

IV. Hierarchy of Dealers: An Extension

In this section we briefly describe a further extension of MeGaDiP and the basic architecture, details of which are our ongoing work. The goal is to further improve the discovery success rate by expanding the searching space of MeGaDiP. However, expanding the searching space will get a large number of dealers involved in the discovery procedure. In order to minimize the number of dealers involved, we suggest a hierarchical architecture to organize the dealers, as shown in Figure 5. Trading domains of 'leaf' dealers are grouped into bigger trading domains, with corresponding parent dealers. The leaf dealers periodically send service advertisement *summaries* (in order to control the volume of service advertisements sent) to their parent dealer. The address of a dealer D's parent dealer can be resolved by performing a DNS lookup of $DNS\_LD(D)$.

The basic media gateway discovery procedure can be

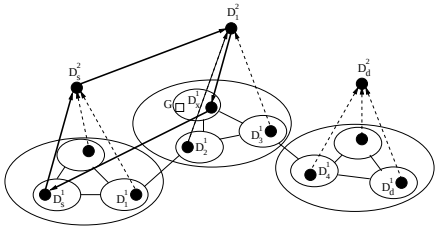

Figure 5: A dealer hierarchy

extended as shown in Figure 5. Suppose the basic discovery procedure terminates at leaf (level 1) dealer D_3^1 without a result, then D_s^1 will forward the query up to its parent dealer D_s^2, and another discovery procedure begins at level 2, and the discovery procedure will terminate at dealer D_1^2. Suppose a qualified gateway G exists in the trading domain of leaf dealer D_x^1, the level 2 dealer D_1^2 will find its service advertisement summary (sent by D_x^1), and D_1^2 will then forward the query down to D_x^1, which will return the full service advertisement of G to D_s^1 as result.

V. Experimental Results

A. Prototype Results

We are implementing a prototype of MeGaDiP in a local testbed, which has three trading domains with

[5]This can be done when the dealer receives a query about the gateway.

[6]We assume that the media streaming mechanism can detect these conditions - for example, by using RTP/RTCP[6].

dealers D_1, D_2, D_3, respectively. In domain I, there is an MPEG video server, and the MPEG-1 video used in our experiment has a run time of 10 minutes and an average data rate of 800Kbps. In domain III, there are 15 video clients. For experimental purpose, each video client only has a low bit rate player in *bitmap* format, with a maximum data rate of 128Kbps. In order to stream the video from the server to the clients, a media gateway with the *MPEG-to-Bitmap transcoding* service is needed. We install the MPEG-to-Bitmap service on four media gateways G_1, G_2, G_3, and G_4, each with the the same background workload. The network connection within each domain is the 10Mbps Ethernet. The network bandwidth between D_1 and D_2 is also 10Mbps, while the bandwidth between D_2 and D_3 is only 1.5Mbps (we generate background traffic to achieve this). The dealers measure the current inter-domain bandwidth every 30 seconds, and the bandwidth consumed by the periodic measurements is only 2Kbps. The media gateways send service advertisements to their local dealers every one minute, *or* when its local resources have run out - whichever is earlier. We perform the following simple

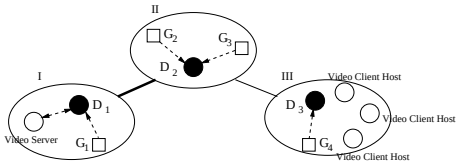

Figure 6: Testbed for MeGaDiP prototype

experiment and obtain some initial results: the video server submits media gateway queries for the 15 clients (sessions) within a period of 5 minutes, and the submission times are uniformly distributed. Table 1 shows: (1) the number of video sessions that result in the discovery of $G_i, i = 1, 2, 3, 4$ by MeGaDiP, and (2) the corresponding (average) response time of MeGaDiP. The values are averaged over 10 trials.

MeGa.	No. of sessions	Response Time
G_1	4.3	20.3ms
G_2	3.5	26.6ms
G_3	3.1	27.2ms
G_4	N/A	N/A

Table 1: Initial Results from Testbed

The response time for the discovery of G_1 is shorter than that for the discovery of G_2 or G_3, because G_1 is in the same domain as the server, and will be discovered first by MeGaDiP. When G_1 runs out of CPU resource, D_1 will forward the queries to D_2, and G_2 or G_3 will be discovered. However, when G_2 and G_3 run out of CPU, D_2 will not forward the query to D_3, because D_2 detects that the current bandwidth from domain II to III is not sufficient to let a *source* MPEG stream get through. All these illustrate MeGaDiP's resource awareness. In addition, notice that the response time for the discovery of G_2 or G_3 is *not* significantly longer than that for G_1, due to MeGaDiP's caching of service advertisement of G_2 or G_3 in D_1.

B. Simulation Results

The local testbed is not adequate to evaluate the performance of MeGaDiP in a wide-area environment. Before the deployment of MeGaDiP on a larger scale, we study its performance by simulation. The trading domains are shown in Figure 7. For simplicity, we assume that there are three hops between any two hosts in neighboring domains, and that the resources are always sufficient for any media gateway discovered. In our first experiment, we

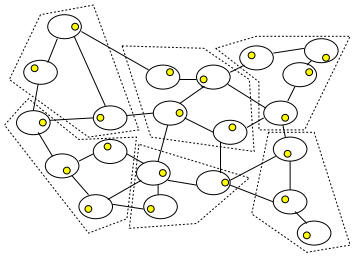

Figure 7: The simulation environment

randomly deploy 8 class I media gateways with a certain service in 8 trading domains. We also randomly choose 200 pairs of <source EH, destination EH>, and the number of hops between each pair of EHs is a multiple of 3. Then we use MeGaDiP, a push-based approach, and a pull-based approach[7] to discover the media gateway between each pair of EHs, respectively. Figure 8 shows the average number of hops along the *source EH - discovered media gateway - destination EH* path. The media gateway discovered by MeGaDiP introduces the least end-to-end delay in term of hop count. Next we evaluate the effect of caching in

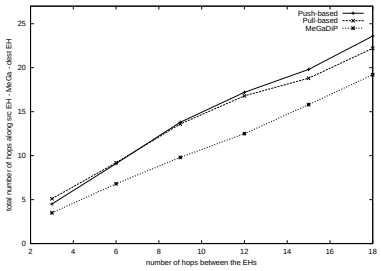

Figure 8: The average number of hops along the *source EH - discovered media gateway - destination EH* path

MeGaDiP. We randomly deploy 10 class I media gateways

[7] In the push-based or pull-based approach, the dealer first looks for a qualified local media gateway. If none is found, it will randomly choose a qualified gateway discovered by the push or pull method in other domains.

with a certain service in 10 trading domains. The EHs pairs are randomly generated at a rate of 100 pairs per minute. We assume that each cached service advertisement will be invalidated within t amount of time, and t is uniformly distributed between 30 and 120 seconds. Figure 9 shows the average number of dealers involved in the discovery for each pair of EHs in every 30-second period. It is obvious from the Figure that caching effectively reduces the number of dealers involved, thus reducing the discovery latency.

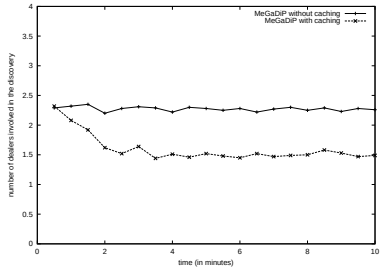

Figure 9: Caching vs. non-caching: the average number of dealers involved in the gateway discovery

VI. Related Work

The Service Location Protocol (SLP) [1] defines the basic common architecture described in this paper. However, SLP was originally designed for service location within one administrative domain, rather than in a wide-area environment. There have been wide-area extensions based on SLP. For example, in [7], a framework for Internet Telephony Gateway (ITG) location is proposed. However, it does not consider the two general classes of media gateways, as defined in MeGaDiP; and it does not have sufficient support for end-to-end resource awareness.

The hierarchy has been proposed as a scalable architecture for wide-area information discovery. Our dealer hierarchy extension is influenced by the Ninja Service Discovery Service (SDS) [2]. The difference between the SDS and the extension of MeGaDiP lies in the forwarding of service queries. In the SDS, if a SDS server can not find qualified service information, the query will be immediately forwarded to the server's parent. In MeGaDiP, the query for a media gateway will first be forwarded to a neighbor dealer at *the same level*. Only when the discovery procedure fails at the current level, will the query be forwarded to the parent of the originating dealer. In fact, the SDS also implicitly assumes RR/SL services. It is easy to show that if the SDS is used in media gateway discovery, the first three problems in Section B still exist.

Resource awareness has been studied in the context of replicated servers selection. For example, in [8], the application-layer anycasting service is proposed to dynamically allocate servers to clients to minimize the response time. A hybrid push/probe technique is used to collect the servers' capability metrics. The server selection problem assume that the candidate servers are already known, and the servers provide RR/SL services. In contrast, MeGaDiP focuses on the discovery of media gateway(s) with sufficient end-to-end resources, and the media gateways do not have to be replicated.

VII. Conclusion

In this paper, we present MeGaDiP, a wide-area media gateway discovery protocol. We first describe a basic architecture common in the current general service location mechanisms. Then we identify the weaknesses of the general service location mechanisms when performing media gateway discovery. Based on the basic architecture, we propose MeGaDiP as a heuristics to be used first when discovering a media gateway. Our initial experimental results demonstrate the feasibility and soundness of MeGaDiP. To further improve the discovery success rate, we also propose an extension of MeGaDiP using a hierarchical dealer architecture. We plan to deploy MeGaDiP in a wider-area environment to study its performance in greater detail. Of particular interest is the impact of the underlying network topology and routing mechanism on the discovery success rate and the validity of the discovery results.

VIII. References

[1] E. Guttman, C. Perkins, J. Veizades, and M. Day "Service Location Protocol, Version 2," *IETF RFC-2165, November 1998*

[2] S. Czerwinski, B. Zhao, T. Hodes, A. Joseph, and R. Katz "An Architecture for a Secure Service Discovery Service," *Proc. ACM/IEEE MobiCom'99, September 1999*

[3] The 2K Team "The 2K Project," *http://choices.cs.uiuc.edu/2K*

[4] A. Downey "Using Pathchar to Estimate Internet Link Characteristics," *Proc. ACM Sigcomm'99, September 1999*

[5] N. Miller and P. Steenkiste "Collecting Network Status Information for Network-Aware Applications," *Proc. IEEE Infocom2000, March 2000*

[6] V. Jacobson, S. Casner, R. Frederick, and H. Schulzrinne "RTP: A Transport Protocol for Real-Time Applications," *IETF Internet Draft, July 1999*

[7] J. Rosenberg and H. Schulzrinne "Internet Telephony Gateway Location," *Proc. IEEE Infocom'98, March 1998*

[8] Z. Fei, S. Bhattacharjee, E. Zegura, and M. Ammar "A Novel Server Selection Technique for Improving the Response Time of a Replicated Service," *Proc. IEEE Infocom'98, March 1998*