Location-based Mobile Querying in Peer-to-Peer Networks

Michel Scholl[1], Marie Thilliez[2*], Agnès Voisard[3]

[1] Cedric – Conservatoire National des Arts et Métiers
75141 Paris Cedex 03 – France
Scholl@cnam.fr
[2] LAMIH – Université de Valenciennes
Le Mont Houy – 59313 Valenciennes Cedex 9 – France
Marie.Thilliez@univ-valenciennes.fr
[3]Fraunhofer ISST and FU Berlin
Mollstr. 1 – 10178 Berlin – Germany
Agnes.Voisard@isst.fhg.de
*this work was done while the author was visiting Cedric lab

Abstract. Peer-to-peer (P2P) networks are receiving increasing attention in a variety of current applications. In this paper, we concentrate on applications where a mobile user queries peers to find either data (e.g., a list of restaurants) or services (e.g., a reservation service). We classify location-based queries in categories depending on parameters such as the user's velocity, the nature of the desired information, and the anticipated proximity of this information. We then propose query routing strategies to ensure the distributed query evaluation on different peers in the application while optimizing the device and network energy consumption.

1 Introduction

Peer-to-peer (P2P) networks are receiving increasing attention in all kinds of applications that range from music file exchange to mobile gaming. One advantage of a P2P principle of distribution is that it allows the sharing of a large quantity of information. To achieve good performance, contextual information should be taken into account. We focus here on applications where (1) information shared by a community is spread over the territory and accessible by static powerful computers (peers) spatially close to this information and (2) mobile users equipped with low energy/light computing facilities look for information close to their current location on the territory. The notion of user context usually encompasses static information such as personal profile but also dynamic information such as his or her location or speed. We here exploit the later. Besides, we utilize some characteristics of the resources to be found, which we denote the query context. In generally distributed frameworks, when a user requests information, the following two modes of interaction are considered: in the *push* mode the system sends him or her "relevant" information and in the *pull* mode the user explicitly asks for information at a certain time instant. This is the focus of our work. A user queries one or more near-by peers to find out about resources. Resources can be data (e.g., a list of restaurants) or services (e.g., a reservation service that obeys constraints like a low cost) represented by tuples in a relational database. Furthermore, each resource is geo-referenced, i.e., it has a point location in the 2D space - a tuple of x and y coordinates. The query is assumed to be a relational query augmented with a proximity spatio-temporal predicate (typically a range query or a nearest neighbor query) which permits to constraint the resources to be found to be in the vicinity of the user. Depending on the query and the strategy, the peers located near-by the one to which the query was issued may try to answer the query and, if necessary, will transmit it in turn to their neighbor peers as long as all these peers are in the user vicinity. Our goal is to classify location-based mobile querying and propose routing strategies in P2P networks that exploit both the environment of the users – their context - and the query context, namely: his or her user mobility materialized by his/her velocity, the type of the desired information (spatial, spatio-temporal, etc.), and the proximity of the desired information.

In distributed database querying, a query has two components which do not necessarily borrow the same path through peers:

1) **Query routing.** A query is issued by the mobile user to initial peer(s), which possibly forward(s) the query to neighbor peers.
2) **Answer routing**. Once a peer has found one answer – i.e., the result of the query is not empty - the latter has to be forwarded to the user. Depending on the strategy, the answer does not necessarily follow the same path through peers as the query itself. In the case of mobile users, the answer forwarding process has to be aware of the user motion model.

This paper is organized as follows. Section 2 gives some background on the considered P2P networks and a simple classification of location-based mobile querying. Section 3 discusses some related work. In Section 4, we specify the parameters of a query, choose three classes of queries and, for each of them, discuss and propose simple query routing strategies, considering both query and answer routing. Section 5 draws our conclusions.

2 Location-based Mobile Querying: a Query Classification

2.1 Architecture

The resources to be shared by several participants called users and possibly mobile are distributed on several static peers (Cf. Figure 1). Peers generally correspond to robust computers. Mobile users, on the other hand, are equipped with handheld devices such as smart phones or PDAs. Peers centralize information about local resources and neighbor connected mobile users. Handheld devices are thin clients employed by users to submit their queries and consult the results. A peer is connected to another one called a *neighbor* peer in function of their networks ranges, particularly wireless networks ranges. In Figure 1, the network ranges are represented by circles.

Handheld devices can communicate not only with peers but with other handheld devices as well, as long as they are in their range. However, our architectural choices are such that a query issued by a handheld device is only evaluated on static peers and not on other handheld devices, therefore reducing communication costs. Due to this distinction of two types of devices, namely handheld devices and peers, our architecture is closer to the so-called hybrid Peer-To-Peer (P2P) architecture [12]. Thanks to its partial data centralization and communication cost reduction, hybrid P2P allows to improve performance and scalability.

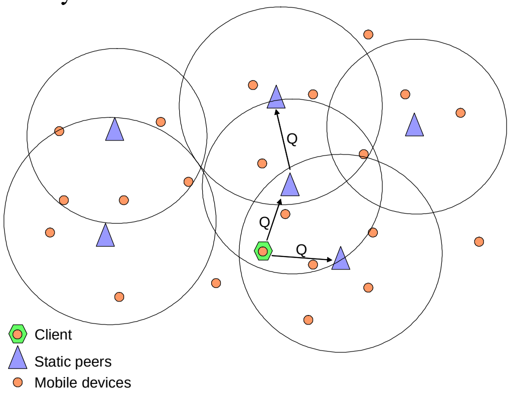

Fig. 1. Representation of the architecture

In the following we assume that (1) information available to the users is structured according to a relational schema common to all peers, and (2) peer *p* stores tuples representing resources spatially close to the peer and possibly has some cached information about its neighbor peers and only those peers.

2.2 Parameters

When considering mobile applications and query routing strategies, many dimensions may be of interest. Let us consider three dimensions:

- The mobility of the client, which can be low or high. If the user is not moving fast or, even, is not moving, we shall say that the user is static or fixed (both terms are used interchangeably in the paper). This case typically corresponds to low network traffic and a user motion so slow that, by the time an answer reaches a user, he or she is still in the range of the peer to which the query was issued. Else we say that the user is mobile.

- The spatial (e.g., near-by drugstores), temporal (e.g., radio shows), or spatio-temporal (e.g., traffic or weather information) type of the requested information. Such a characteristic of the queried resources is important as the user who issues a query has a location and issues the query at a given time instant. Moreover, the proximity and time validity predicates in the query formulation have an impact on its answer.
- Another parameter of interest in many applications is the cardinality of the answer over the network, i.e., the possible rarity of the information. For instance, a user maybe looking for a particular friend (who is unique) or for a store that has musical scores from the XVIIIth century (*rare* and accessible only from a few peers). In contrast, gas stations are *frequent* and accessible from a large number of peers. Usually, when one is looking for a rare resource, space and time restrictions are loose or even not an issue.

Table 1 gives a query classification based on these three parameters.

Table 1. Example queries for various clients and resources and rarity of the solutions

User\Resource	**Fixed**		**Mobile**	
	Rare	**Frequent**	**Rare**	**Frequent**
Fixed	***Q1a -1:*** Find a picture of *this* monument ***Q1a- 2:*** Find a musical library with scores from the XVIIIth century	***Q1b:*** Find the picture of this monument taken by Rosa Newton	***Q3a:*** Find friend Joe	***Q3b:*** Find a taxi near my hotel
Mobile	**Rare**	**Frequent**	**Rare**	**Frequent**
	Q2a: Find a Breton restaurant as I walk	***Q2b:*** Find a gas station as I drive	***Q4a :*** Find friend Joe as we both drive	***Q4b :*** Find a taxi as I walk

3 Related Work

Related to routing strategies in wireless and mobile P2P environment, [13,1,4] are examples of solutions to improve search in P2P networks. However, these solutions are dedicated to wired P2P network such as Internet. In our environment, the users are mobile and querying is based on the location of the information. In particular, user mobility implies that we have to consider both the dynamicity of environment due to heterogeneity, unreliability, mobility, and so on, and the limited energy of handheld devices of the mobile users.

Location-dependent queries (LDQ) [7] consider the geographic localization of mobile users in the query evaluation process. Different solutions can be used to localize the user, such as GPS [3] or wireless network based solutions [9]. Some recent studies present solutions to evaluate LDQs in mobile environments. Many LDQ evaluation solutions are dedicated to centralized environments such as cellular networks [10]. Other solutions are

based on distributed environments [8], however, these solutions do not consider different mobility profiles in the query evaluation (e.g., pedestrian, car driver, and so on.). Other studies concern the evaluation of LDQs using caching techniques [2].
In [11], various challenging issues in mobile querying in a distributed environment are generally presented. Compared to this work, ours presents a classification of wireless queries and focuses on routing strategies to evaluate LDQs. Solutions in query routing for wired environments exist [5,6], however, to the best of our knowledge, none of them considers the user mobility and the energy constraints of mobile devices.

4 Query routing strategies

This section, proposes three routing strategies as a function of the user mobility profile and the rarity of requested information, with the following restrictions:

- We do not consider queries where a mobile user is looking for a mobile resource: all resources that a user is querying are static and associated with a unique peer.
- While there exist a variety of spatial predicates, such as: resources within a rectangle, within a disc of radius ε (range query), closest resource ("Nearest Neighbor" or NN query), or *k* closest resources (k-NN query), we restrict our attention to range queries.
- We assume that a query asks for resources to be found immediately, i.e. at the query issue time. Queries whose result is to be found in the future lead to other strategies which take into account the anticipation of the query. We leave them as a future work.
- In the following, we omit querying resources that are only valid in some time intervals or at some time instants. We believe that the extension of the proposed strategies to temporal (news) and spatio-temporal predicates (near-by sportive events, theater plays) taking into account the temporal type of the resource is straightforward.

A query is specified by the following tuple: $Q=[id_c, id_q, sql, \varepsilon, t_0, t_{max}, traj]$.
Query Q with identifier id_q, issued at time t_0 by a mobile client c with identifier id_c and trajectory *traj* is interpreted as: "Return all resources satisfying query *sql* within a disc of radius ε, prior to time t_{max}".
Peer *p* chooses a strategy that depends on the rarity of the resource (we assume that rarity is an attribute of one of the associated relations, but it can be inferred by the system as well), t_{max} and the speed of the client whose components are specified in parameter *traj*. Depending on the speed of *c*, and the rarity of the resource, one out of a number of strategies is chosen. We consider the three following cases:

- ***fixed-to-fixed (FtoF)*:** the client speed is low, t_{max} is high, and the traffic is light enough so as to guarantee that peer *p* has enough time to find the answer(s) (among its resources or by its neighbor peers) and forward it back to client *c* which, in the meantime, has not left zone *Z* controlled by *p*. Q1* (Table 1) is an example of ***FtoF*** query.

- ***mobile-rare (MR)*:** the speed/time max/traffic configuration is such that the client is likely to have left zone *Z* at time *t in* $]t_0,t_{max}]$, once peer *p* - which has an answer *a* - wants to forward it back to *c*. One typical consideration would be: high value of t_{max} and high speed. Since they are rare, all answers found must be forwarded to the client wherever he or she stands at time *t'>t.* If the found resource is itself mobile, then its trajectory at time *t* is forwarded as well to *c*. In Table 1, there are 2 examples of ***MR*** queries : Q2a or Q4a.
- ***mobile-frequent (MF)*:** as for the MR case, the client has a good chance for having left the zone Z, however, here it is not worth forwarding the query since the searched resource has a large probability of being found in other peers along *c* trajectory. The simple strategy chosen here is to forward the query to peers close to *c* trajectory; if peer *p'* at time *t* has an answer it sends it back to *c* if *c* is in peer *p'* zone at time *t*. Queries Q2b and Q4b (Table 1) are ***MF*** queries.

Although there exist several strategies for each of the aforementioned scenarios, we give an algorithm for only one in each case. It is further assumed that each peer *p* locally stores the distance to its furthest resource d_{max} as well as to each of its neighbor peers *p'* the furthest resource distance d_{max}*'* to *p'*. A refined variant which would require more space would be to store the maximum distance to peer *p* (same for its neighbors) for each of the relations. No further assumption is made on the zones covered by a peer. In each peer, *d(p)* stores for peer *p* the maximum distance to any resource registered in *p* (*p* is either the peer hosting the table or any of its neighbors). *distance(a,b)* denotes the Euclidean distance between two points *a* and *b*.

4.1 Strategy FtoF

The strategy (functions *FtoFquery* and *FtoFanswer*) works as follows. *Q* is issued by *c* to peer p_0 (*c* belongs to the zone associated with p_0*)*. If *distance(c,*p_0*)* is larger than $\varepsilon + d(p_0)$ then no resource under *p* satisfies the query. Else *p* runs the query and checks for each answer *a* under its zone, whether its (Euclidean) distance to *c* is less than ε. It also sends query *q* to its neighbor peers *p'* whose distance to *c* is less than $\varepsilon + d(p')$ and waits for its answers. If a neighbor *p'* has in turn another neighbor peer *p"* such that $distance(c,p") \leq \varepsilon + d(p")$, then query Q is forwarded to *p"*. *p"* answers if any are returned to *p'*, etc. Depending on the traffic, a variant strategy would choose to send back the answer directly to the initial peer p_0. Note that in the applications we target, in which resources are to be looked for in the vicinity of the client, the expected length of the chain of visited peers should be small (one or two). Eventually, peer p_0 has collected all answers and sends them back to the client. *FtoFanswer(A,p)* forwards the answers *A* back to peer *p*. The initial call is *FtoFquery(Q,*t_0*,c,*p_0*)*.

```
Algorithm FtoFquery (Q,t,p,p')
Input: Q=[ $id_c$ , $id_q$ , sql , ε , $t_0$ , $t_{max}$ , traj , ],
            p'=current peer (to which the query has been sent)
            p=The peer that sent the query
Output: A= set of answer tuples
A:= emptyset
If $t>t_{max}$ then FtoFanswer(A,p)
Else if distance(c,p’) > $\varepsilon$ +d(p’) then FtoFanswer(A,p)
    Else For each a in sql
            If distance(a, c) $\leq \varepsilon$ then A+={a}
    For each neighbor peer p”, A+={FtoFquery(Q, t, p’, p”)}
    FtoFanswer(A,p)
```

4.2 Strategy MR

Strategy MR works as follows. *Q* is issued by *c* like in strategy FtoF (Function *MRQuery*). If the current peer does not have an answer, Q is forwarded to its neighbor peers. Else each peer p having found a resource at time $t_l<t_{max}$ forwards it back according to the following strategy (algorithm *MRanswer*). Given the trajectory, *p* estimates the client location l_θ =[x,y] after a time θ. θ is an estimator of the time necessary to reach the mobile client. θ depends on the client speed *s*, the level of traffic, and the distance *d* between the peer and the client at time t_1. A simple estimator would be $\theta = kd^\gamma s^\beta$ where *k* depends on the traffic. Let Δ(p,p’) be the line between *p* and a neighbor peer. $\Delta(p,l_\theta)$ is the line between p and the mobile client at time θ. Let α{ Δ1, Δ2 } denote the angle between the 2 lines in argument. Then peer *p* chooses as a neighbor peer (*p’*) to which the query is to be forwarded the one such that $\alpha\{\Delta(p,p'), \Delta(p,l_\theta)\}$is minimum. Then when *p’* receives the answer, either the client is inside the zone covered by *p’*, and the answer is forwarded back to the client, or the answer has to be forwarded to *p”*. *p”* is chosen according to the same algorithm. However, we have two variants: either compute the position according to the same value of θ or refresh θ and thus the future location of the client. The initial call is *MRquery* (Q,t_0,p_0). Function *estim* estimates $l_{\theta.}$

```
Algorithm MRquery (Q,t,p’)
Input: Q=[ $id_c$ , $id_q$ , sql , $\varepsilon$ , $t_0$ , $t_{max}$ , traj , ],
            p’=current peer (to which the query has been sent)
Output: A= set of answer tuples
A:= emptyset
If $t>t_{max}$ then Return
Else if distance(c,p’) > $\varepsilon$ +d(p’) then Return
    Else For each a in sql
            If distance(a ,c) $\leq \varepsilon$ then A+={a}
    If A non empty then begin
                    min:=360; $\theta = kd^\gamma s^{\beta;}$ $l_\theta$ := estim(traj,$t_0$, s, θ)
                    For each neighbor peer p if $\alpha\{\Delta(p, p'), \Delta(p', l_\theta)\}$<min
                            then begin
                                min = $\alpha\{\Delta(p, p'), \Delta(p, l_\theta)\}$
                                $p_1$:=p
                            end
```

```
                        MRanswer(A,$p_1$,Q)
                        end
                    Else For each neighbor peer p”, A+={MRquery(Q,t,p”)}
```

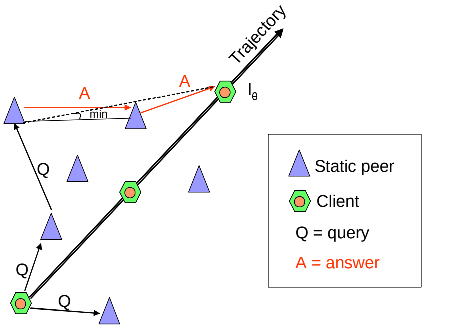

Fig. 2. Query and answer routing in strategy MR

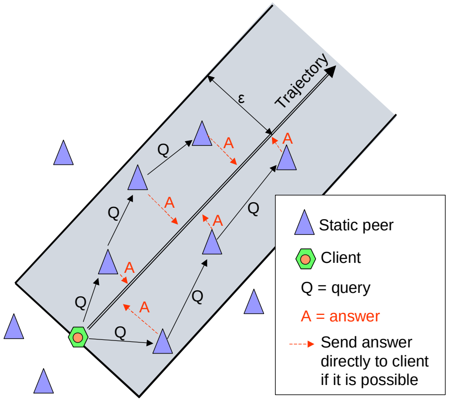

Fig. 3. Query and answer routing in strategy MF

```
Algorithm MRanswer(A,p’,Q)
If $t>t_{max}$ then return
    Else if  c in zone(p) then return (A,c)
        Else begin
            min:=360; $\theta = kd^{\gamma}s^{\beta}$; $l_\theta$ := estim(traj,$t_0$, s, θ)
            For each neighbor peer p if α{ Δ(p,p’), Δ(p’,$l_\theta$ )}<min
                                    then begin
                                        min = α{ Δ(p,p’), Δ(p,$l_\theta$ )}
                                        $p_1$:=p
                                        end
            MRanswer(A,$p_1$,Q)
            end
Return
```

4.3 Strategy MF

The resources are found as in *MRQuery* (function *MFquery*). However, in contrast to strategy MR, possible results are numerous and ubiquitous. Then, one minimizes traffic and energy consumption related to communications between peers by not forwarding results if the client is out of the spatial scope. The searched resources can be found later on, on the mobile user trajectory. In the following variant, if peer *p* has an answer and if client *c* is still in the spatial range of *p*, then the answer is forwarded back to *c*. Else (*c* not in range anymore) the query is forwarded to neighbor peers close to c trajectory. Function *inScope* in Algorithm *MFQuery* returns true if the peer specified as a parameter is in the semi-rectangle illustrated in Figure 3. This semi-rectangle represents the spatial scope of the query as a function of the client trajectory and ε. In another “continuous query”, to obtain all possible results, variant peer *p* would forward the query to neighbor peers anyway (even if the answer was successfully sent back to *c*). A third variant would be to choose a strategy similar to that of *MRanswer* for forwarding the query more adapted to cases where the client is far away once a query in a peer has failed.

```
Algorithm MFquery (Q,t,p’)
Input:  Q=[ $id_c$ , $id_q$ , sql , $\varepsilon$ , $t_0$ , $t_{max}$ , traj , ],
          p’=current peer (to which the query has been sent)
Output: A= set of answer tuples
A:= emptyset
If  $t>t_{max}$ then Return
   Else begin
         if distance(c,p’) < $\varepsilon$ +d(p’) then
             begin
              For each a in sql
                       If distance(c,a) $\leq \varepsilon$ then A+={a}
              M := false;
              If A non empty then M := MFanswer(A,p”,Q)
              If (M= false and t< $t_{max}$) then
                  For each neighbor peer p”,
                         If inScope(p”, traj, t, $\varepsilon$) then MRquery(Q,t,p”)
              end
         end
```

```
Algorithm MFanswer (Q,t,p’)
M := false
If $t>t_{max}$ then return (M)
   Else If  c in zone(p) then
                        begin
                          Forward (A,c); M:=true; return (M)
                        end
                        Else return(M)
```

5 Conclusion

In this paper, we presented a simple classification of location-based mobile queries in P2P networks according to context variables of both the user and the query, namely the client mobility, the spatio-temporal nature of the query, and its rarity. These parameters have an impact on both query routing and answer routing. We chose three classes of queries and described for each of them a detailed query and answer routing strategy. As a future work we intend to evaluate the performance of these strategies by comparing them to other variants among which some were suggested earlier. Among the parameters that impact on the performance, the level of traffic (number of clients and number of queries) as well as the client speed are noteworthy.

Acknowledgments. We wish to thank the anonymous reviewers for their comments. This work was carried out in the framework of a PROCOPE French-German grant.

6 References

[1] M. Demirbas, H. Ferhatosmanoglu. Peer-To-Peer Spatial Queries in Sensor Networks. In *Proc. 3rd Intl. IEEE Conference on Peer-to-Peer Computing (P2P'03),* Computer Society Press, Los Alamitos, CA, USA, 2003.

[2] T. Doulkeridis, V. Zafeiris, and M. Vazirgiannis. The Role of Caching in P2P Service Discovery. In *Proc. Intl. ACM Conference on Mobile Data Management (MDM)*, ACM Press, New York, N.-Y, 2005.

[3] B. Hofmann-Wellenhof, H. Lichtenegger, and J. Collins. *Global Positioning Systems: Theory and Practice*, *5th edition*. Springer Verlag, Berlin/Heidelberg/New York, 2001.

[4] S. K. Goel, M. Singh, D. Xu, B. Li. Efficient Peer-To-Peer Data Dissemination in Mobile Ad-Hoc Networks. In *Proc. Intl. Workshop on Ad Hoc Networking (IWAHN, in conjunction with Intl. Proc. Intl. Conference on Parallel Processing, ICPP),* 2002.

[5] M. Iles, D. Deugo. A Search for Routing Strategies in a Peer-To-Peer Network Using Genetic Programming. In *Proc. IEEE Symposium on Reliable Distributed Systems (SRDS)*, Computer Society Press, Los Alamitos, CA, USA. 2003.

[6] E. Michlmayr, S. Graf, W. Siberski, W. Nejdl. Query Routing with Ants. In *The Semantic Web: Research and Applications, Second European Semantic Web Conference*, A. Gomez-Perez, J. Euzenat (Eds.), Lecture Notes in Computer Science No. 3532, Springer Verlag, Berlin/Heidelberg/New York, 2005.

[7] J. Schiller, A. Voisard. *Location-Based Services*. Morgan Kaufmann/Elsevier, San Francisco, CA, USA, 2004.

[8] M. Thilliez and T. Delot. Evaluating Location Dependent Queries Using ISLANDS. In *Advanced Distributed Systems: Third International School and Symposium, (ISSADS)*, Lecture Notes in Computer Science No. 3061, Springer Verlag, Berlin/Heidelberg/New York, 2004.

[9] M. Thilliez, T. Delot, S. Lecomte. An Original Positioning Solution to Evaluate Location-Dependent Queries in Wireless Environments. *Journal of Digital Information Management - Special Issue on Distributed Data Management*, (3) 2, 2005.

[10] J. Xu, D.K. Lee. Querying Location-dependent Data in Wireless Cellular Environment. In *Proc. W3C Workshop on Position Dependent Information Services*, 2000.

[11] B. Xu, O. Wolfson. Data Management in Mobile Peer-To-Peer Networks. In *Proc. of the 2nd Intl. Workshop on Databases, Information Systems, and Peer-to-Peer Computing (DBISP2P'04),* Lecture Notes in Computer Science No. 3367, Springer Verlag, Berlin/Heidelberg/New York, 2004.

[12] B. Yang, H. Garcia-Molina. Comparing Hybrid Peer-To-Peer Systems. In *Proc. of the 27th Int. Conf. On Very Large Data Bases*, Morgan Kaufmann/Elsevier, San Francisco, CA, USA. 2001.

[13] B. Yang, H. Garcia-Molina. Improving Search in Peer-To-Peer Networks. In *Proc of IEEE Intl. Conference on Distributed Computing Systems (ICDCS)*, Computer Society Press, Los Alamitos, CA, USA, 2002.