RKNN Using Clipping Methodologies

[1] M.Vinothkumar, [2]Dr.K.Selvam
[1]*Research Scholar,* [2]*Professor*
Dept of Computer Applications
Dr.MGR Educational & Research Institute, Chennai, Tamil Nadu, India.

Abstract -Reverse k Nearest Neighbor (RKNN) query is proposed based on the k Nearest Neighbor (KNN) query, which can be used to evaluate the influence of the query objects. At present RKNN query algorithms are mostly based on static or continuous objects, The algorithms for efficiently answering queries about large populations of moving objects are gaining interest. Given a group of nearby space objects as the query input, the authors propose the continuously monitoring RKNN algorithms based on the half-space pruning, compute the minimum bounding rectangles (MBRs) containing the query objects and consider the objects in the rectangles as a whole. Then, In order to get the query's final RKNN results.

Keywords:- Half-space, Dominance, Metric Based, Probabilistic, Prune.

INTRODUCTION

The clipping for RNN query processing in spatial databases has been well studied it is non-trivial to devise clipping strategies for RNN query processing on uncertain data. For example, if we na¨ıvely use every instance of a Filtering Object to perform bi- sector clipping , it will a huge computation cost due to large number of instances in each uncertain object. Instead, we devise non-trivial generalization of bisector bisector clipping for minimum bounding rectangles (MBRs) of uncertain objects based on a novel notion of normalized half space. Verification is extremely expensive in probabilistic RNN query processing because, in order to verify an object as probabilistic RNN, we need to take into consideration not only the instances of this object but also the instances of query object and other nearby objects. Hence it is important to devise efficient clipping rules to reduce the number of objects that need verification.

Types of clipping

In this, we present several clipping rules from the following orthogonal perspectives:

- Half space based clipping that exploits geometrical properties
- Dominance based clipping that exploits topological properties
- Metric based clipping
- Probabilistic clipping that exploits the probability threshold

We remark that the first three of the above clipping techniques are the same as those presented. The only deference is that, in this chapter, we present generalized versions of these clipping rules that can be applied on multidimensional space (and not only on 2d space).

Half Space Clipping

Consider a query point q and a filtering object U that has n instances {u1 , u2, . . . , un}. Let Hu :q be the half space between q and ui. Any instance u ∈/ U that lies in ∩n Hu :q has zero probability to be the RNN of q because by the property of Hui:q , u is closer to every ui than to q. where the bisectors between q1 and the instances of A are drawn and the half spaces Ha1 :q1 and Ha2 :q1 are shown. Intersection of the two half spaces is shown shaded and any point that lies in the shaded area is closer to both a1 and a2 than q1. For this reason, b2 cannot be the RNN of q1 in any possible world. This clipping is very expensive because we need to compute intersection of all half spaces Hui:q for every ui ∈ U . Below we present our clipping rules that utilize the MBR of the entire filtering object, Rf il , to prune the candidate object with respect to a query instance q or the MBR of uncertain query object Q.

Clipping using Rf il and an instance q

First we present the intuition. Consider the example of Fig. 5.3 where we know that the point p lies on a line M N but we do not know the exact location of p on this line. The bisectors between q and the end points of the line (M and N) can be used to prune the area safely.

In other words, any point that lies in the intersection of half spaces HM :q and HN :q (grey area) can never be the RNN of q. It can be proved that whatever be the location of point p on the line M N , the half space Hp:q always contains HM :q ∩ HN :q. Hence any point p′ that lies in HM :q ∩ HN :q would always be closer to p than to q and for this reason cannot be the RNN of q.

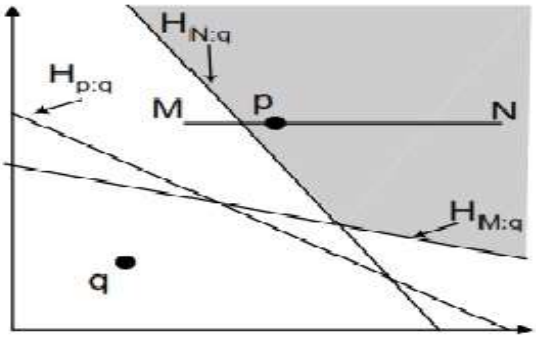

Fig 1: - The exact location of the Point p on line M N is not known

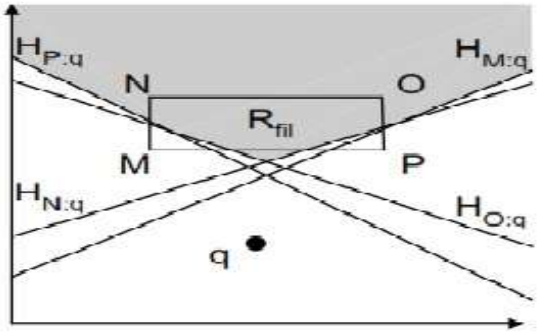

Fig 2: - Any point in shaded area can not be RNN of q in any possible world

Based on the above observation, below we present a clipping rule for the case when the exact location of a point p is unknown within some hyper-rectangle R*fil*.

Clipping Rule 5.3.2 : Let Rf il be a hyper-rectangle and q be a query point. For any point p that lies in T2d i=1 HCi:q (Ci is the it*h* corner of Rfil), dist(p, q) > maxdist(p,Rfil) and thus p cannot be the RNN of q.

Note that this clipping rule is a generalized version of the clipping rule 3.3.4 presented in Chapter 3. The proof is similar to the proof of clipping rule 3.3.4 and is omitted. Consider the example of Fig. 2. Any point that lies in shaded area is closer to every point in rectangle Rf il than to q. Note that if Rf il is a hyper rectangle that encloses all instances of the filtering object Ui then any instance u ∈ Uj,j=i that lies in T2HCi:q can never be the RNN of q in any possible world.

Clipping using Rf il and RQ

Clipping rule 5.3.2 prunes the area such that any point lying in it can never be the RNNof some instance q. However, the points in the pruned area may still be the RNNs of other instances of the query. Now, we present a clipping rule that prunes the area using Rfil and RQ such that any point that lies in the pruned area cannot be the RNN of any instance of Q.

Consider the example of Fig. 5.5 where the exact location of the query point q on line M N is not known. Unfortunately, in contrast to the previous case of Fig. 1, the bisectors between p and the end points of the line M N do not define the area that can be pruned. If we prune the area Hp:M ∩ Hp:N (the grey area), we may miss some point p′ that is the RNN of q. Fig. 3 shows a point p′ that is the RNN of q but lies in the shaded area. This is because the half space Hp:q does not contain Hp:M ∩ Hp:N . This makes the clipping using Rf il and RQ challenging.

Note that if Hp:N is moved such that it passes through the point where Hp:q intersects Hp:M then Hp:M ∩ Hp:N would be contained by Hp:q . We note that in the worst case when p lies infinitesimally close to point M , Hp:q and Hp:M intersect each other at point c which is the centre of line joining p and M . Hence, in order to safely prune the area, the half space Hp:N should be moved such that it passes through the point c. The point c is shown in Fig 3.

A half space that is moved to the point c is called a normalized half space and a half space Hp:N that is normalized is denoted as H′. Fig. 3 shows H′ p:N in broken line and Hp:N ∩ Hp:M (the dotted shaded area) can be safely pruned.

Before we present our clipping rule for the general case that uses 2d half spaces to prune the area using hyper-rectangles RQ and Rf il , we define the following concepts:

Antipodal Corners: Let C be a corner of rectangle R1 and C′ be a corner in R2, the two corners are called antipodal corners2 if for every dimension i where C[i] = R1L[i] then C′[i] = R2H [i] and for every dimension j where C[j] = R1H [j] then C′[j] = R2L [j]. Fig. 5.6 shows two rectangles R1 and R2. The corners D and O are antipodal corners. Similarly, other pairs of antipodal corners are (B, M), (C, N) and (A, P).

Antipodal half space: A half space that is defined by the bisector between two antipodal corners is called antipodal half space. Fig. 4. shows two antipodal half spaces 2 RL[i] (resp. RH [i]) is the lowest (resp. highest) coordinate of a hyper-rectangle R in it*h* dimension HM :B and HP :A.

Normalized half space: Let B and M be two points in hyper-rectangles R1 and R2, respectively. The normalized half space H′ is a space defined by the bisector between M and B that passes through a point c such that c[i] = (R1L[i] + R2L [i])/2 for all dimensions i for which B[i] > M [i] and c[j] = (R1H [i] + R2H [j])/2 for all dimensions j for which B[j] ≤ M [j]. Fig.6 shows two normalized (antipodal) half spaces H′ and H′.The point c for each half space is also shown. The inequalities 1 and 2 define the half space HM :B and its normalized half space H′ respectively.

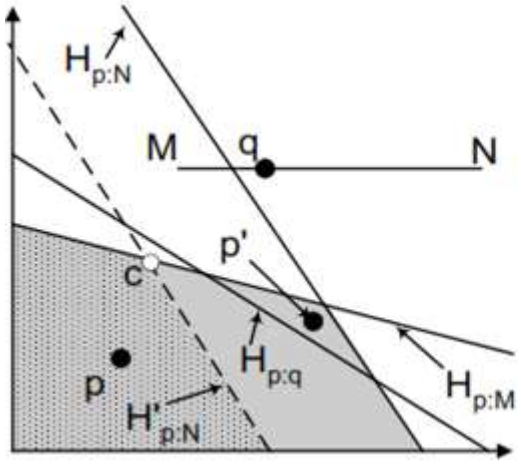

Fig 3: - Any point in dotted area can never be RNN of q

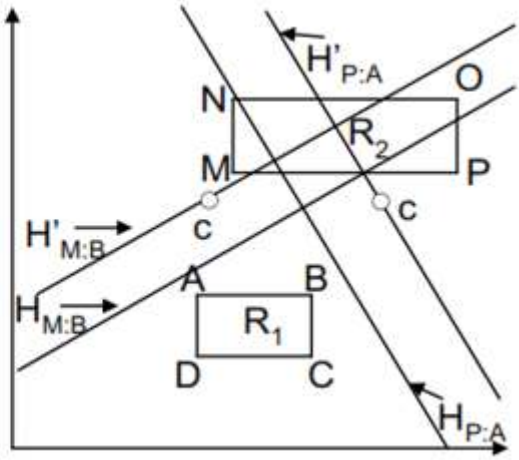

Fig 4: - Antipodal corners and normalized half spaces

Clipping Rule 5.3.3 : Let RQ and R*fil* be two hyper-rectangles. For any point p that lies di= H′, mindist(p, RQ) > maxdist(p, R*f il*) where H′i iis normalized half space between C (the it*h* corner of the rectangle Rf il) and its antipodal corner C′ in RQ.

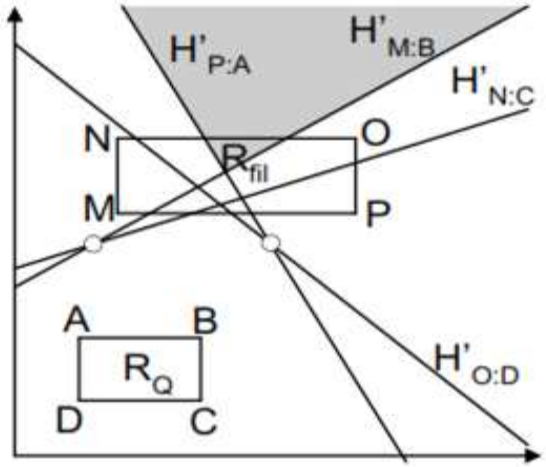

Fig 5: - Any point in Shaded area can never be RNN of q £ Q

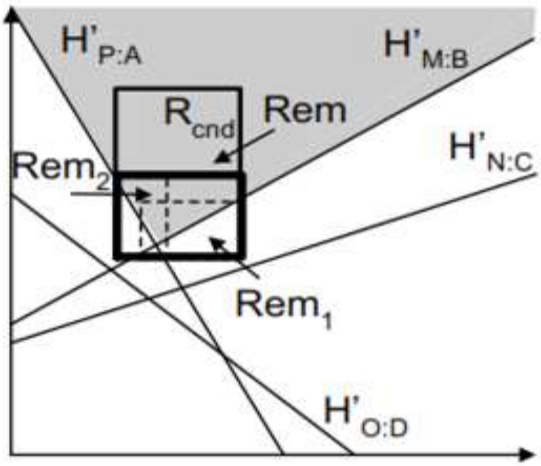

Fig 6: - Clipping part of the candidate object R_{end} that cannot be pruned

Consider the example of Fig. 5 where the normalized antipodal half spaces are drawn and their intersection is shown shaded. Any point that lies in the shaded area is closer to every point in rectangle Rf il than every point in rectangle RQ. Note that if Rf il and RQ are the MBRs enclosing all instances of an uncertain object Ui and query object Q, respectively, any instance u ∈ Uj,j=i that lies in the pruned d region, T2 H′ , cannot be RNN of any instance of q Q in any possible world. Even if the clipping region

partially overlaps with Rfil, we can still trim the part of any other hyper-rectangle that falls in the pruned region.

It is known that exact trimming becomes inefficient in high dimensional space, therefore, we adopt the loose trimming of Rcnd proposed.

The overall half space clipping algorithm that integrates clipping rules 5.3.2 and 5.3.3 is illustrated in Algorithm 7. For each half space, we use the clipping algorithm in [013] to find a remnant rectangle Remi ⊆ Rcnd that cannot be pruned (lines 4 and 7).

Algorithm 1 : hspace-clipping (Q, Rf il , Rcnd)

Input: Q: an MBR containing instances of Q ; Rf il : the MBR to be used for trimming

R_{cnd}: the candidate MBR to be trimmed

Description:

1: Rem = ∅ /* Remnant rectangle */

2: for each corner Ci of Rf il do

3: if Q is a point then

4: Remi = clip(Rcnd , HCi :Q)/* clipping algorithm [013] */

5: else if Q is a hyper-rectangle then

6: C′ = antipodal corner of Ci in Q

7: Remi = clip(Rcnd , H′ i)/* clipping algorithm [013] */

8: enlarge Rem to enclose Remi

9: if Rem = Rcnd then

10: return Rcnd

11: return Rem

After all the half spaces have been used for clipping, we calculate the MBR Rem ⊆ Rcnd as the minimum bounding hyper rectangle covering every Remi. As such, we trim the original Rcnd to Rem. For better illustration we zoom Fig. 5 and show the clipping of a hyper-rectangle Rcnd in Fig. 6.

The algorithm returns Rem1, Rem2 (rectangles shown with broken lines) when H′and H′ are parameters to the clipping algorithm, respectively. For the half spaces H′and H′ the whole hyper-rectangle Rcnd can be pruned so the algorithm eturns . The remnant hyper-rectangle Rem is an MBR that encloses Rem1 and Rem2. Note that at any stage if the remnant rectangle Rem becomes equal to Rcnd, the clipping by other bisectors is not needed so Rcnd is returned without further clipping (line 10).

Dominance Clipping

We first give the intuition behind this clipping rule. The clipping by using clipping rule 5.3.3 in two dimensional spaces. The normalized half spaces are defined such that if Rf il is fully dominated3 by RQ in all dimensions then all the normalized antipodal half spaces meet at point Fp We also observe that for the case when Rf il is fully dominated by RQ, the angle between the half spaces that define the pruned area (shown in grey) is always greater than 90◦ . Based on these observations, it can be verified that the space dominated by Fp (the dotted-shaded area) can be pruned.

Let RQ be the MBR containing instances of Q. We can obtain the 2d regions. RUi be an MBR of a filtering object Rfil that lies completely in one of the 2d regions. Let f be the furthest corner of RU from RQ and n be the nearest corner of RQ from f . The frontier point Fp lies at the centre of line joining f and n.

Clipping Rule 5.3.4

Any instance u ∈ Uj that is dominated by the frontier point Fp of a filtering object cannot be RNN of any q ∈ Q in any possible world.

The dominance clipping (one in each region). In each partition the shaded area is dominated by Fp and can be pruned. Note that if Rf il is not fully dominated by RQ, we cannot use this clipping rule because the normalized antipodal half spaces in this case do not meet at the same point.

For example, the four normalized antipodal half spaces intersect at two points in Fig. 5. In general, the clipping power of this rule is less than that of the half space clipping. The area pruned by the half space clipping (shaded area) and dominance clipping (dotted area). The main advantage of this clipping rule is that the clipping procedure is computationally more efficient than the half space clipping, as checking the dominance relationship and trimming the hyper-rectangles is easier.

Metric Based Clipping

Clipping Rule 5.3.5 An uncertain object Rcnd can be pruned if maxdist(Rcnd, Rf il) <mindist(Rcnd, RQ). This clipping approach is the least expensive. Note that it cannot prune part of Rcnd, i.e., it either invalidates all the instances of Rcnd or does nothing.

Probabilistic Clipping

Note that we did not discuss probability threshold while presenting previous clipping rules. In this section, we present a clipping rule that exploits the probability threshold and embeds it in all previous clipping rules to increase their clipping powers.

A simple exploitation of the probability threshold is to trim the candidate object using previous clipping rules and then prune the object if the accumulative appearance probability of instances within its remnant rectangle is less than the threshold.

Next, we present a more powerful clipping rule that is based on estimating an upper bound of the RNN probability of candidate objects. In previous clipping rules, we prune some area using MBR of a query object RQ and a filtering object Rfil.

We observe that the area pruned by using R′ and R′ always contains the area pruned by RQ and Rfil where $R' \subseteq RQ$ and $R' \subseteq Rfil$. The shaded area is pruned when R′ and Rf il are used for clipping and the dotted shaded area is pruned when RQ and Rfil are used. Note that this observation also holds for the dominance clipping.

We can use the observation presented above to prune the objects that cannot have RNN probability greater than the threshold. First, we give a formal description of this clipping rule and then we give an example.

Clipping Rule 5.3.6: Let the instances of Q be divided into n disjoint sets {Q1, Q2 , ..., Qn } and RQi be the minimum bounding rectangle enclosing all instances in Qi. Let {Rcnd1,Rcnd2, ..., Rcndn } be the set of bounding rectangles such that each Rcndi contains the instances of the candidate object that cannot be pruned for Qi using any of the clipping rules. Let PRQi and PRcndi be the total appearance probabilities of instances in Qi and Rcnd, respectively. If Pn(PRcndi · PRQi), the candidate object can be pruned.

Clipping rule 5.3.6: Computes an upper bound of the RNN probability of the candidate object by assuming that all instances in Rcndi are RNNs of all instances in Qi. The candidate object can be safely pruned if this upper bound is still less than the threshold.

Example 5.3.7 : MBRs of the query object RQ and a candidate object Rcnd along with their instances (q1 to q5 and u1 to u4). Assume that all instances within an object have equal appearance probabilities (e.g: pqi = 0.2 for every qi and pui = 0.25 for every ui). Suppose that no part of Rcnd can be pruned using RQ and any filtering object Rfil (for etter illustration, filtering object is not shown).

We prune Rcnd using the rectangle RQ1 that is contained by RQ. This trims Rcnd and the remnant rectangle R1 is obtained. Similarly, R2 is the remnant rectangle when clipping rules are applied for RQ2. Note that only the instances in R1 (u1 and u2) can be the RNN of instances in RQ1 (q3, q4 and q5). Similarly, no instance can be the RNNs of any instance in RQ2 because R2 is empty. So the maximum RNN probability of Rcnd is $(0.6 \times 0.5) + (0.4 \times 0) = 0.3$. If the probability threshold is greater than 0.3, we can prune Rcnd. Otherwise, we can continue to trim Rend by using the smaller rectangles contained in RQ1 .

In our implementation, we build an R-tree on query object and the clipping rule is applied iteratively using MBRs of children. For more details, please see Algorithm 11. Although the smaller rectangles R′contained in Rfil can also be used, we do not use them because unlike query object there may be many filtering objects.

Hence, using the smaller rectangles for each of the filtering objects would make this clipping rule very expensive in practice (more expensive than the efficient verification presented.

Integrating the clipping rules

Algorithm 8 is the implementation of clipping rules 5.3.2 to 5.3.5. Specifically, we apply clipping rules in increasing order of their computational costs (i.e., from clipping rule 5.3.5 to 5.3.2). While simple clipping rules are not as restricting as more expensive ones, they can quickly discard many non-promising candidate objects and save the overall computational time.

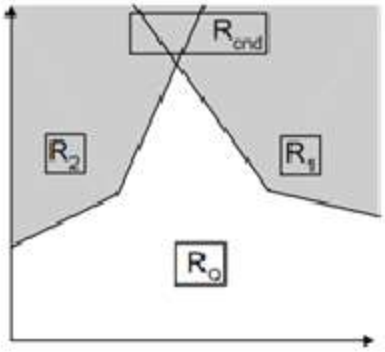

Fig 7 :- R_{end} can be pruned by R_1 and R_2

It is important to use all the filtering objects to filter a candidate objects. Rcnd cannot be pruned by either R1 or R2 , but will be pruned by considering both of them. Two subtle optimizations in the algorithm are:

- If mindist(Rcnd, Rfil) > maxdist(RQ, Rcnd) for a given MBR Rfil , then Rfil cannot prune any part of Rcnd. Hence such Rf il is not considered for dominance and half space clipping (lines 4-5). However, Rf il may still prune some other candidate objects, so we remove such Rfil only from a local set of filtering object, Sfil. This optimization reduces the cost of dominance and half space clipping.
- If the frontier point Fp1 of a filtering object Rf il1 is dominated by the frontier point Fp2 of another filtering object Rf il2 , then Fp1 can be removed from Sf il because the area pruned by Fp1 can also be pruned by Fp2 . However, note that a frontier point cannot be used to prune its own rectangle. Therefore, before deleting Fp1, we use it to prune rectangle belonging to Fp2 . This optimization reduces the cost of dominance clipping.

Algorithm 8 : Prune(Q, Sf il , Rcnd)

Input: RQ: an MBR containing instances of Q ; Sf il : a set of MBRs to be used for

trimming Rcnd: the candidate MBR to be trimmed

Description:

1: for each Rf il in Sf il do

2: if maxdist(Rcnd, Rf il) < mindist(RQ, Rcnd) then /* Clipping rule 5.3.5 */

3: return

4: if mindist(Rcnd, Rf il) > maxdist(RQ, Rcnd) then

5: Sf il = Sf il − Rf il /* Rf il cannot prune Rcnd */Rem = Rcnd

6: for each Rf il in Sf il do

7: if Rf il is fully dominated by RQ in a partition p then /* Clipping rule 5.3.4*/

8: if some part of Rem lies in the partition p then

9: Rem = the part of Rem not dominated by Fp

10: if (Rem =) then return

11: for each Rf il in Sf il do

12: Rem = hspace clipping(RQ, Rf il , Rem) /* Clipping Rules 5.3.2 and 5.3.3 */

13: if (Rem =) then return

14: return Rem

CONCLUSION

We formalize probabilistic reverse nearest neighbor query that is to retrieve the objects from the uncertain data that have higher probability than a given threshold to be the RNN of an uncertain query object. We develop an efficient algorithm based on various novel clipping approaches that solves the probabilistic RNN queries on multidimensional uncertain data. The experimental results demonstrate that our algorithm is even more efficient than a sampling-based approximate algorithm for most of the cases and is highly scalable and performance of algorithm is tested by employing real data set. With a group of neighboring space objects as inquire point set, the experiment result shows that the proposed algorithm's efficiency is better than direct algorithm. This algorithm also has practical application value, could be used to evaluate the set object influence. Future work will focus on improving algorithm, researching R*K*NN and high-dimensional RKNN.

REFERENCES

[1] Tao Yu-fei,Papadias D,Lian X,et al.Multidimensional reverse kNN search. The VLDB Journal The International Journal on Very Large Data Bases[J]. 2007,PP293-415

[2]Wu Wei,Yang Fei,Chan Yong,Tan Kian-Lee. FINCH:Evaluating reverse k nearest neighbor queries on location data. Proceedings of the Very Large Data bases[J].2008,PP1056-1067

[3]Wang Wei,Wang Wenping.An algorithm for finding the smallest circle containing all points in a given point set. Journal of Software[J].2000,PP1237-1240

[4] Kyriakos Mouratidis, Dimitris Papadias, Spiridon Bakiras and Yufei Tao. A Threshold-Based Algorithm for Continuous Monitoring of k Nearest Neigh- bors. TKDE, pages 1451–1464, 2005.

[5] Yufei Tao, Reynold Cheng, Xiaokui Xiao, Wang Kay Ngai, Ben Kao and Sunil Prabhakar. Indexing Multi-Dimensional Uncertain Data with Arbi- trary Probability De7nsity Functions. In VLDB, pages 922–933, 2005.

[6]Korn F, Muthukrishnan S. Influence sets based on reverse nearest neighbor queries. Special Interest Group of Data[J]. 2000, PP201-212

[7]Wu Wei,Chee Fei Yang,Chan Yong,Tan Kian-Lee. Continuous reverse k nearest neighbor monitoring. Proceedings of the 9th International Conference on Mobile Data Management[J].2008,PP132-139.