DEIM Forum 2017 E3-5

不確実データベースにおけるMaxRSクエリ処理手法

中山　侑紀† 　天方　大地† 　原　隆浩†

† 大阪大学大学院情報科学研究科マルチメディア工学専攻　〒 565–0871 大阪府吹田市山田丘 1–5
E-mail: †{nakayama.yuki,amagata.daichi,hara}@ist.osaka-u.ac.jp

あらまし　本稿では，不確実データベースにおける MaxRS（Maximizing Range Sum）クエリについて考える．不確実データベースは，複数のインスタンスおよびその生起確率の組で表現されるデータで構成されるデータベースである．MaxRS クエリは，スコアをもつ空間データの集合およびユーザが指定した長方形の大きさが与えられたとき，その長方形に包含されるデータのスコアの和が最大となる長方形の位置を検索するものである．これまでに，不確実データベースを想定した MaxRS 検索に関する研究が存在しないため，本稿で，Probabilistic MaxRS（P-MaxRS）クエリを新たに定義する．P-MaxRS クエリは，MaxRS となる確率が 0 でない位置およびその確率の組の集合を取得するものであるが，既存研究の MaxRS クエリ処理アルゴリズムを単純に適用した場合，非効率的である．そこで，P-MaxRS クエリの解となる位置を限定し，その後，限定した位置が MaxRS となる確率を効率的に計算するアルゴリズムを提案する．実データおよび人工データを用いた実験により，提案アルゴリズムの有効性を示す．

キーワード　不確実データベース，空間データ，MaxRS クエリ

1. はじめに

近年，GPS などの位置情報取得技術の発展や位置情報サービスの普及に伴い，空間データベースへの関心が高まっている．また，スマートフォンなどのモバイル端末が普及したことにより，空間データが爆発的に増加しているため，大量のデータから重要な情報を検索するクエリ処理に関する研究が盛んに行われている [10], [12]．空間データベースに対するクエリ処理の 1 つとして，MaxRS（Maximizing Range Sum）クエリが提案されている [5]．MaxRS クエリは，スコアを持つ空間データの集合およびユーザが指定した長方形の大きさが与えられたとき，その長方形に包含されるデータのスコアの和が最大となる長方形の位置を検索するものである．以下の例を用いて，MaxRS クエリを説明する．

例 1.1. 図 1 は MaxRS クエリの一例を表している．最も外側の太線の長方形は空間データの存在範囲を表しており，点線および実線の長方形，および黒点は，それぞれユーザが指定した大きさの長方形，および空間データを表している．簡単化のため，空間データのスコアは全て 1 とする．この例では，黒点を最も多く包含している実線の長方形の位置が MaxRS クエリの解（の一つ）である．

MaxRS クエリは，長方形の大きさを指定するだけで重要な位置を検索できるため，多くのアプリケーションにとって有用である．以下に，MaxRS クエリのアプリケーション例を示す．

例 1.2. 観光客にとって魅力的な観光場所を提示するシステムを考える．空間データとして，観光地の位置およびその観光地の評価点で構成されるデータを想定する．一般的に，観光客がある一定期間（例えば，一日）に移動できる範囲は限られているため，その範囲を MaxRS クエリで指定する長方形の大きさとする．このとき，MaxRS クエリにより，観光客が移動できる範囲内に存在する観光地の評価点の和が最も高い場所を検索できる．つまり，このシステムでは，観光客が最も満足できると考えられる場所を提示できる．

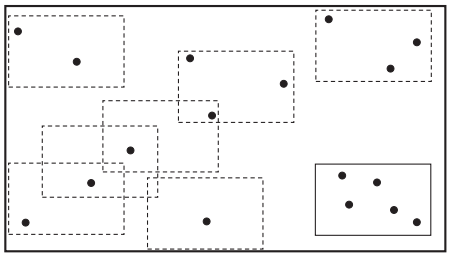


図 1　MaxRS クエリの一例

例 1.2 以外のアプリケーション例として，センサネットワークがある．例えば，道路上に設置したセンサにより，交通量を測定している場合を考える．MaxRS クエリは，最も交通量が多い位置を検索できる．この検索結果は，交通量が多い場所にレストラン等を出店したい企業にとって有用である．

上記のように有用なアプリケーションが存在するため，これまでに MaxRS クエリに関する研究は数多く行われている [2], [3], [5], [6], [8], [11], [14]．しかし，これらの研究は，データの不確実性を考慮していない．例えば，例 1.2 の場合，観光地は様々な人により評価されており，それらを同時に考えたとき，ある観光地の評価点は，複数の値とその値の生起確率の組で表現される．そこで，本稿では不確実性が存在するデータに対する MaxRS（Probabilistic MaxRS，P-MaxRS）クエリの効率的な処理方法を提案する．P-MaxRS クエリは，MaxRS となる確率が 0 でない位置とその確率の組を検索するクエリである．単純には，空間データの取り得る値の組合せ全てに対して，既存の MaxRS クエリ処理アルゴリズムを適用することによって P-MaxRS クエリを処理できる．しかし，空間データの取り

得る値の組合せの数は，データ数に対して指数的に増加するため，データ数が多い場合，非効率的である．そこで，P-MaxRS クエリの解となる位置（データ集合）を限定した後，限定したデータ集合が MaxRS となる確率を効率的に計算するアルゴリズムを提案する．また，実データおよび人工データを用いた実験から，提案アルゴリズムがクエリ処理時間を削減できていることを確認した．

以下では，2. で P-MaxRS クエリを定義し，提案アルゴリズムで利用する既存アルゴリズムを説明する．また，3. で提案アルゴリズムについて述べる．4. で実データおよび人工データを用いた実験により，提案アルゴリズムの性能を評価する．5. で関連研究について述べ，最後に，6. で本稿をまとめる．

2. 事 前 準 備

本稿では，不確実データベースにおける MaxRS クエリについて考える．まず，本稿で考えるデータモデルおよび問題を詳細に定義し，その後，提案アルゴリズムで利用する既存研究のアルゴリズムを紹介する．

2.1 データモデル

不確実データベース． 不確実データベース U は，n 個の不確実オブジェクトで構成される（$U = \{u_1, u_2, \cdots, u_n\}$）．以降，曖昧性が生じない限り，不確実オブジェクトを単にオブジェクトと記述する．オブジェクト u_i は，m 個のインスタンスで構成される（$u_i = \{u_{i,1}, u_{i,2}, \cdots, u_{i,m}\}$）．また，インスタンス $u_{i,j}$ は，二次元の位置，スコア，および生起確率で構成される（$u_{i,j} = \langle loc, s, p\rangle$）．本稿では，オブジェクトの位置は決定的であると想定する（$u_i.loc = u_{i,1}.loc = u_{i,2}.loc = \cdots = u_{i,m}.loc$）．また，オブジェクト u_i のインスタンス中に同じスコアは存在しない（$u_{i,1}.s \neq u_{i,2}.s \neq \cdots \neq u_{i,m}.s$）．さらに，インスタンスのスコアは自然数（$u_{i,j}.s \in \mathbb{N}$）とし，各オブジェクトを構成するインスタンスの生起確率の和は 1（$\sum_{u_{i,j} \in u_i} u_{i,j}.p = 1$）と想定する．このような不確実データベースの定義は，属性レベルモデルと呼ばれ，不確実データベースの分野では一般的である [1]．また，本稿では，全ての属性値が決定的であるオブジェクトで構成されるデータベースを，単にデータベースと記述し，不確実データベースと区別する．

Possible World． ある Possible World $\mathcal{W}$ は，不確実データベース U 中の各オブジェクトからインスタンスを 1 つずつ選択したものである．つまり，ある Possible World $\mathcal{W} \in \Omega$ は，$\mathcal{W} = \{u_{1,\alpha}, u_{2,\beta}, \cdots, u_{n,\gamma}\}$ である．ここで，Ω は，U から生成される全ての Possible World の集合である．また，$\mathcal{W}$ の生起確率を $\Pr(\mathcal{W}) = \prod_{u_{i,j} \in \mathcal{W}} u_{i,j}.p$ とする．以上の定義より，$|\Omega| = m^n$，および $\sum_{\mathcal{W} \in \Omega} \Pr(\mathcal{W}) = 1$ である．

2.2 問 題 定 義

MaxRS 確率． 不確実データベース U，および長方形の大きさを指定したクエリ q が与えられる．クエリ q で指定された大きさの長方形に包含される（ただし，長方形の回転は考慮しない）オブジェクトの集合を R^q をする．考えられる全ての R^q の集合を $\mathcal{R}^q$ としたとき，$R^q \in \mathcal{R}^q$ の MaxRS 確率を以下の式で定義する．

$$\Pr[MaxRS_q = R^q] = \sum_{\mathcal{W} \in \Omega} \Pr(\mathcal{W}) \cdot \delta(MaxRS_q^{\mathcal{W}} = R^q)$$

$$\delta(MaxRS_q^{\mathcal{W}} = R^q) = \begin{cases} 1, & \text{if } R^q = \operatorname*{argmax}_{R \in \mathcal{R}^q} f^{\mathcal{W}}(R) \\ 0, & \text{otherwise} \end{cases}$$

$$f^{\mathcal{W}}(R) = \sum_{u_i \in R, u_{i,j} \in \mathcal{W}} u_{i,j}.s \tag{1}$$

$f^{\mathcal{W}}(R)$ は，ある Possible World $\mathcal{W}$ における，オブジェクト集合 R 内の全てのオブジェクトのスコアの和である．よって，$\delta(MaxRS_q^{\mathcal{W}} = R^q)$ は，$\mathcal{W}$ において，オブジェクト集合 R^q が MaxRS であるなら 1，MaxRS でないなら 0 となる関数である．ここで，本稿では，任意の Possible World $\mathcal{W}$ において，MaxRS となるオブジェクト集合と同じスコアをもつオブジェクト集合は存在しないものとする．よって，$\sum_{R^q \in \mathcal{R}^q} \Pr[MaxRS_q = R^q] = 1$ である．しかし，上記のようなオブジェクト集合が存在する場合でも，同じスコアであるオブジェクト集合のうちのいずれか 1 つのみを MaxRS とすることで，$\sum_{R^q \in \mathcal{R}^q} \Pr[MaxRS_q = R^q] = 1$ は成り立つ．

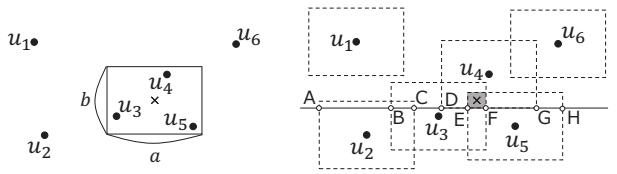
MaxRS 確率を用いて，以下に Probabilistic MaxRS（P-MaxRS）クエリを定義する．

定義 2.1（P-MaxRS クエリ）． 不確実データベース U および長方形の大きさを指定したクエリ q が与えられる．このとき，P-MaxRS クエリの解は，$\mathcal{R}^q$ 中で MaxRS 確率が 0 でないオブジェクト集合およびその MaxRS 確率の組 $\langle R, \Pr[MaxRS_q = R]\rangle$ の集合 $\mathcal{ANS}$ である．

ユーザの要求によっては，P-MaxRS クエリの解を取得した後，MaxRS 確率が閾値以上であるオブジェクト集合や MaxRS 確率が上位 k 番目までのオブジェクト集合のみを取得することもできる．

2.3 既存研究のアルゴリズム

文献 [9] において，RI（Rectangle Intersection）クエリを効率的に処理するアルゴリズムが提案されている．RI クエリは，大きさが等しい長方形集合が与えられたとき，最も多くの長方形が重なっている領域を求めるものである．また，オブジェクト集合を長方形集合に変換し，長方形の重みを考慮した RI クエリ処理を行うことで，MaxRS クエリの解を求められる [5]．図 2 より，RI クエリの解である影の付いた領域（図 2(b)）が MaxRS クエリの解の長方形の中心（× 印）を包含していることが分かる．また，文献 [1] において，不確実データベースにおける ALL_SUM クエリを効率的に処理するアルゴリズムが提案されている．ALL_SUM クエリは，スコアをもつ不確実オブジェクト集合が与えられたとき，取り得る全てのスコアの和およびその確率の組を求めるクエリである．提案アルゴリズムにおいて，文献 [9] で提案されているアルゴリズム（以降，PS），および文献 [1] で提案されているアルゴリズム（DP_PSUM2）を利用するため，本節でこれらのアルゴリズムについて簡単に説明する．



(a) MaxRS クエリ　　(b) RI クエリ

図 2　MaxRS クエリの変換例

PS. PS は，平面走査法を基にしたアルゴリズムである．まず，オブジェクト集合と長方形の大きさが与えられたとき，与えられたオブジェクト集合中の各オブジェクトの位置を中心とする与えられた大きさの長方形集合に変換する．また，その長方形集合を RI クエリの入力とする．その後，水平線を y 座標の最小値から最大値まで走査することにより，RI クエリを処理する．具体的には，(i) 水平線が長方形の下辺に達したとき，および (ii) 水平線が長方形の上辺に達したときにそれぞれ以下の操作を行う．(i) 下辺から構成されるインターバルを 2 分木に挿入し，2 分木内のインターバルのスコアを更新する．さらに，これまでのスコアの最大値を超える場合，そのインターバルを途中解とする．(ii) 上辺から構成されるインターバルを 2 分木から削除する．

図 2(b) を用いて，インターバルのスコアの更新を説明する．図 2(b) は，水平線が u_4 を変換した長方形の下辺に達したときを示している．全ての長方形の重みを 1 としたとき，インターバル AB，BC，CD，DE，EF，FG，および GH のスコアはそれぞれ 1，2，1，2，3，2，および 1 である．よって，インターバル EF を途中解とする．以上より，水平線が y 座標の最大値に達したとき，最大のスコアを持つインターバルが解と分かる．PS は，オブジェクト数が n のとき，2 分木に対してインターバルの挿入および削除をそれぞれ n 回行う．そのため，計算量は $\mathcal{O}(n \log n)$ である．

DP_PSUM2. DP_PSUM2 は動的計画法を基にしたアルゴリズムである．不確実オブジェクト集合 $U = \{u_1, u_2, \cdots, u_n\}$ が与えられたとする．このとき，逐次的にオブジェクトを追加し，そのスコアの和および確率を計算する．以下に，この計算方法を示す．

$$ps(s,1) = \begin{cases} u_{1,j}.p, \text{ if } \exists j \text{ such that } u_{1,j}.s = s \\ 0, \text{ otherwise} \end{cases} \quad (2)$$

$$ps(s,i) = \sum_{k=1}^{m} ps(s - u_{i,k}.s, i-1) \cdot u_{i,k}.p, \text{ if } i > 1 \quad (3)$$

$ps(s,i)$ は，i 個のオブジェクト $u_1, u_2, \cdots, u_i$ で構成されるオブジェクト集合のスコアの和が s である確率を表す．まず，式 (2) により，オブジェクト u_1 のみを考慮したときの取り得るスコアおよび確率を計算する．その後，式 (3) により，$ps(\cdot, i-1)$ を参照して，オブジェクト u_i を追加したときのスコアの和と確率 $ps(\cdot, i)$ を計算する．表 1 および図 3 を用いて，具体的に説明する．

表 1　不確実オブジェクト集合 U の例

オブジェクト	インスタンス（⟨ スコア, 確率 ⟩）
u_1	$u_{1,1} = \langle 1, 0.5 \rangle, u_{1,2} = \langle 2, 0.5 \rangle$
u_2	$u_{2,1} = \langle 2, 0.3 \rangle, u_{2,2} = \langle 3, 0.7 \rangle$
u_3	$u_{3,1} = \langle 1, 0.2 \rangle, u_{3,2} = \langle 2, 0.8 \rangle$

s＼i	1 ($\{u_1\}$)	2 ($\{u_1, u_2\}$)	3 ($\{u_1, u_2, u_3\}$)
1	0.5	0	0
2	0.5	0	0
3	0	0.15	0
4	0	0.5	0.03
5	0	0.35	0.22
6	0	0	0.47
7	0	0	0.28

図 3　DP_PSUM2 の計算例

例 2.1. 表 1 は不確実オブジェクト集合 U を表している．また，図 3 は，DP_PSUM2 により U に対する ALL_SUM クエリを処理する計算過程（$ps(s,i)$）を表している．まず，u_1 のみを考慮した不確実オブジェクト集合 $U = \{u_1\}$ の取り得るスコアおよび確率を計算する．その結果が $ps(1,1) = 0.5$ および $ps(2,1) = 0.5$ である．次に，不確実オブジェクト集合に u_2 を追加する（$U = \{u_1, u_2\}$）．このとき，$ps(\cdot, 1)$ を利用して，$ps(\cdot, 2)$ を計算する．具体的には，u_2 を追加することによって，$ps(3,2) = ps(1,1) \cdot u_{2,1}.p = 0.15$，$ps(4,2) = ps(1,1) \cdot u_{2,2}.p + ps(2,1) \cdot u_{2,1}.p = 0.5$，また $ps(5,2) = ps(2,1) \cdot u_{2,2}.p = 0.35$ となる．u_3 を追加したときも同様の計算を行うことにより，図 3 の結果が得られる．このとき，ALL_SUM クエリの解は，s および $ps(s,3)$ の組のリストである．

DP_PSUM2 は，1 つのオブジェクトを追加したとき，取り得るスコアの数（N_s）× インスタンス数（m）回の計算を行う．また，n 個のオブジェクトを逐次的に追加するため，計算量は $\mathcal{O}(N_s \cdot m \cdot n)$ である．

3. 提案アルゴリズム

本章では，P-MaxRS クエリを効率的に処理するアルゴリズムについて説明する．まず，提案アルゴリズムの概要について述べ，その後，提案アルゴリズムを詳細に説明する．

3.1 提案アルゴリズムの概要

まず，(I) P-MaxRS クエリの解となるオブジェクト集合を取得する．2.2 節で定義した $\mathcal{R}^q$ は MaxRS 確率が 0 であるオブジェクト集合を含んでいる．そこで，PS および PS を拡張したアルゴリズムにより，MaxRS 確率が 0 であるオブジェクト集合を除外する．ここで除外されずに残ったオブジェクト集合の集合を $\mathcal{AR}$ とする．図 4 は，$\mathcal{AR}$ の例である．長方形および黒点は，それぞれ $\mathcal{AR}$ 中のオブジェクト集合 R_i および R_i 中のオブジェクトを表しており，白点は $\mathcal{AR}$ に含まれないオブジェクトを表している．その後，(II) $\mathcal{AR}$ 中の各オブジェクト集合の MaxRS 確率を計算する．一般的には，図 4 のように，1 つのオブジェクトが複数のオブジェクト集合に含まれる場合

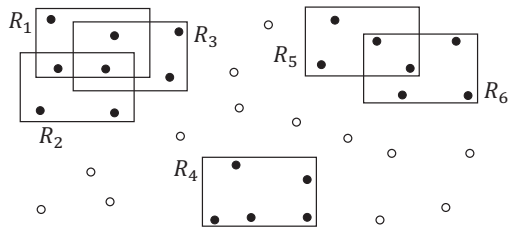
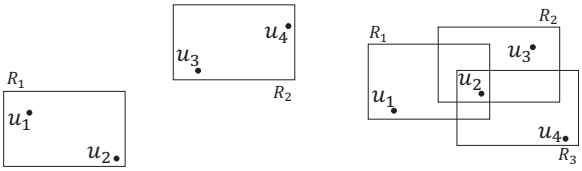


図 4　オブジェクト集合の集合 $\mathcal{AR}$ の例

Algorithm 1: CalculateAR(U, q)

Input: 不確実データ集合 U，長方形の大きさ $q = (a, b)$
Output: オブジェクト集合の集合 $\mathcal{AR}$

1 $O_{min} = \emptyset$，$O_{max} = \emptyset$，$I = \emptyset$
2 **for** $\forall u_i \in U$ **do**
3 　$O_{min} = O_{min} \cup \langle u_i.loc, u_i.MinScore \rangle$
4 $I \leftarrow$PS(O_{min})
5 $lb_{score} = I.score$
6 **for** $\forall u_i \in U$ **do**
7 　$O_{max} = O_{max} \cup \langle u_i.loc, u_i.MaxScore \rangle$
8 $\mathcal{AR} \leftarrow \mathrm{PS}_{\mathrm{extend}}(O_{max}, lb_{score})$
9 **return** $\mathcal{AR}$



(a) 共有オブジェクトが存在しない場合　(b) 共有オブジェクトが存在する場合

図 5　オブジェクト集合 U' の例

表 2　図 5(b) 中のオブジェクト

オブジェクト	インスタンス（⟨ スコア, 確率 ⟩）
u_1	$\langle 5, 1.0 \rangle$
u_2	$\langle 9, 0.5 \rangle, \langle 4, 0.5 \rangle$
u_3	$\langle 6, 1.0 \rangle$
u_4	$\langle 7, 1.0 \rangle$

表 3　図 5(b) の DP_PSUM2 の結果

オブジェクト集合	取り得るスコアと確率（⟨ スコア, 確率 ⟩）
R_1	$\langle 14, 0.5 \rangle, \langle 9, 0.5 \rangle$
R_2	$\langle 15, 0.5 \rangle, \langle 10, 0.5 \rangle$
R_3	$\langle 16, 0.5 \rangle, \langle 11, 0.5 \rangle$

がある．この場合，DP_PSUM2 をそのまま適用できない．そこで，提案アルゴリズムでは，$\mathcal{AR}$ をオブジェクトを共有しているオブジェクト集合同士の集合（独立集合）に分割し，各オブジェクト集合が属する独立集合中での MaxRS 確率（ローカル MaxRS 確率）を計算する．その後，ローカル MaxRS 確率を基に，各オブジェクト集合のスコアを比較することにより，各オブジェクト集合の MaxRS 確率を計算する．単純には，独立集合に含まれるオブジェクトから生成される Possible World 毎に MaxRS となるオブジェクト集合を計算することでローカル MaxRS 確率を求められる．しかし，Possible World の数は，独立集合に含まれる全てのオブジェクトの数に対して指数的に増加する．そこで，部分的に DP_PSUM2 を適用することにより，計算しなければならない Possible World の数を削減する．

3.2　解となるオブジェクト集合の取得

アルゴリズム 1 に，P-MaxRS クエリの解となるオブジェクト集合を取得するアルゴリズムを示す．まず，オブジェクト集合が MaxRS となる場合に必要なスコア（lb_{score}）を取得する．そのために，各オブジェクトのスコアを取り得るスコアの最小値（$MinScore$）として，PS を実行する（1–4 行）．各オブジェクトのスコアは 1 つ（最小値）であるため，PS をそのまま実行できる．また，PS の解のスコアを lb_{score} とする（5 行）．このとき，lb_{score} は，PS の解であるオブジェクト集合のスコアの和の最小値である．そのため，オブジェクト集合のスコアの和の最大値が lb_{score} より小さい場合，そのオブジェクト集合の MaxRS 確率は 0 である．よって，オブジェクト集合のスコアの和の最大値が lb_{score} 以上となるオブジェクト集合の集合 $\mathcal{AR}$ を取得する．まず，各オブジェクトのスコアを取り得るスコアの最大値（$MaxScore$）とする（6–7 行）．その後，PS の拡張アルゴリズム（$\mathrm{PS}_{\mathrm{extend}}$）を実行する（8 行）．$\mathrm{PS}_{\mathrm{extend}}$ は，PS と同様に平面走査を実行し，途中解のスコアが lb_{score} 以上であるオブジェクト集合を全て取得する．ただし，$R \subset R'$ であるオブジェクト集合 R は $\mathcal{AR}$ に含めない．そのため，$\mathrm{PS}_{\mathrm{extend}}$ では，$\mathcal{AR}$ 中に $R \subset R'$ となるオブジェクト集合 R' が存在しない場合に，オブジェクト集合 R を $\mathcal{AR}$ に含める．

3.3　各オブジェクト集合の MaxRS 確率の計算

式 (1) より，単純には，オブジェクト集合 $U' = \cup R$（$R \in \mathcal{AR}$）（図 4 の黒点）から生成される Possible World 毎に PS アルゴリズムを実行することで，各オブジェクト集合の MaxRS 確率を計算できる．しかし，Possible World の数は $m^{|U'|}$ であるため，$|U'|$ が大きい場合，非効率的である．そこで，オブジェクト集合が取り得るスコアの和（以降，オブジェクト集合のスコア）を比較することにより，各オブジェクト集合の MaxRS 確率を計算する．複数のオブジェクト集合に共通しているオブジェクト（共有オブジェクト）が存在しない場合（図 5(a)），$\mathcal{AR}$ 中の各オブジェクト集合 R に DP_PSUM2 を適用し，R のスコアとその確率を計算する．その後，それらのスコアを比較することにより，R の MaxRS 確率を計算できる．一方，共有オブジェクトが存在する場合（図 5(b)），上述した方法では MaxRS 確率を計算できない．

図 5(b)，表 2，および表 3 により，これを説明する．図 5(b) 中のオブジェクトは，表 2 中のインスタンスで構成されているものとする．簡単化のため，共有オブジェクト u_2 以外のオブジェクトのインスタンスは 1 つとする．このとき，各オブジェクト集合に DP_PSUM2 を適用した結果が表 3 である．表 3 より，オブジェクト集合のスコアを比較したとき，全てのオブジェクト集合は MaxRS である場合が存在する．例えば，R_1 のスコアが 14，R_2 のスコアが 10，R_3 のスコアが 11 のとき，R_1

Algorithm 2: DivideARintoIP($\mathcal{AR}$)

Input: $\mathcal{AR}$
Output: 独立集合の集合 $\mathcal{IP}$

```
1  IP_1 ← R_1 ∈ AR
2  IP ← IP_1
3  for ∀R_i ∈ AR \ R_1 do
4      newFlag = 1
5      for ∀IP ∈ IP do
6          if ∃R_j ∈ IP|R_i ∩ R_j ≠ ∅ then
7              IP = IP ∪ R_i
8              newFlag = 0
9              break
10     if newFlag = 1 then
11         IP_new ← R_i
12         IP = IP ∪ IP_new
13 return IP
```

が MaxRS である．しかし，R_1 のスコアが 14 である Possible World を考えた場合，u_1，u_2，u_3，および u_4 のスコアは，それぞれ 5，9，6，および 7 である．そのため，スコアが 16 である R_3 が MaxRS となり，R_1 は MaxRS でない．

このように，全ての Possible World の集合 Ω 中に存在しないインスタンスの組合せを考えているため，各オブジェクト集合に DP_PSUM2 を適用し，スコアを比較する方法は，オブジェクト集合の MaxRS 確率を正確に計算できない．そのため，共有オブジェクトが存在する場合，Possible World を考慮しなければならない．そこで，提案アルゴリズムでは，Possible World の基となるオブジェクト数を削減する．まず，独立集合を定義する．

定義 3.1（独立集合）．$\mathcal{AR}$ 中の全てのオブジェクト集合を重複しないように以下の 2 つの条件を満たす集合 $IP_1, IP_2, \cdots$ へ分割する．

$$\forall R_i \in IP_a, \exists R_j \in IP_a \setminus R_i, R_i \cap R_j \neq \emptyset$$
$$\forall R_i \in IP_a, \forall IP_b \in \mathcal{IP} \setminus IP_a, \forall R_j \in IP_b, R_i \cap R_j = \emptyset$$

ここで，$\mathcal{IP} = \{IP_1, IP_2, \cdots\}$ である．このようなオブジェクト集合の集合 IP_a を独立集合と呼ぶ．

アルゴリズム 2 に，$\mathcal{AR}$ を独立集合に分割するアルゴリズムを示す．まず，初期化処理として，オブジェクト集合 1 つで構成される IP_1 を $\mathcal{IP}$ に挿入する（1–2 行）．次に，各オブジェクト集合 R_i を，共有オブジェクトをもつオブジェクト集合 R_j が属する独立集合に追加する（6–9 行）．R_i が既に存在する独立集合に挿入されなかった場合，自身を新たな独立集合とし，$\mathcal{IP}$ に追加する（10–12 行）．図 4 の $\mathcal{AR}$ は，アルゴリズム 2 により，$IP_1 = \{R_1, R_2, R_3\}$，$IP_2 = \{R_4\}$，$IP_3 = \{R_5, R_6\}$ に分割される．次に，オブジェクト集合のローカル MaxRS 確率について考える．

ローカル MaxRS 確率．オブジェクト集合 R が属する独立集合 IP において MaxRS となる確率をローカル MaxRS（LMaxRS）

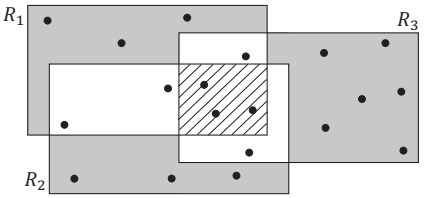


図 6　独立集合の一例

確率と定義する．R のスコア $S(R)$ が s である場合の R の LMaxRS 確率は以下の式で求められる．

$$\Pr[LMaxRS_q = R, S(R) = s] = \sum_{\mathcal{W} \in \Omega_{IP}} \Pr(\mathcal{W}) \cdot \delta(LMaxRS_q^{\mathcal{W}} = R) \cdot \delta(S(R) = s)$$

$$\delta(LMaxRS_q^{\mathcal{W}} = R) = \begin{cases} 1, & \text{if } R = \underset{R' \in IP}{\operatorname{argmax}} f^{\mathcal{W}}(R') \\ 0, & \text{otherwise} \end{cases}$$

$$\delta(S(R) = s) = \begin{cases} 1, & \text{if } s = f^{\mathcal{W}}(R) \\ 0, & \text{otherwise} \end{cases}$$

ここで，Ω_{IP} は，オブジェクト集合 $U_{IP} = \cup R$（$R \in IP$）（図 6 の全ての黒点）から生成される全ての Possible World の集合である．

オブジェクト集合の MaxRS 確率．オブジェクト集合のローカル MaxRS 確率を用いて，オブジェクト集合 $R_i \in IP_a$ の MaxRS 確率を以下の式で計算する．

$$\Pr[MaxRS_q = R_i] = \sum_{s \geqq lb_{score}} \{\Pr[LMaxRS_q = R_i, S(R_i) = s] \prod_{IP_b \in \mathcal{IP} \setminus IP_a} (\sum_{R_j \in IP_b} \sum_{s' < s} \Pr[LMaxRS_q = R_j, S(R_j) = s'])\} \quad (4)$$

$\sum_{R_j \in IP_b} \sum_{s' < s} \Pr[LMaxRS_q = R_j, S(R_j) = s']$ は，$R_i \notin IP_b$ である独立集合 IP_b 中の任意のオブジェクト集合がスコア $s'(< s)$ でローカル MaxRS となる確率の和を意味している．

オブジェクト集合のローカル MaxRS 確率の計算．単純には，オブジェクト集合 U_{IP} から生成される全ての Possible World において，独立集合 IP 内で MaxRS となるオブジェクト集合を計算することで，オブジェクト集合のローカル MaxRS 確率を求められる．しかし，Possible World の数は，$|U_{IP}|$ に対して，指数的に増加する．そこで，提案アルゴリズムでは，DP_PSUM2 を部分的に利用することで，Possible World の基となるオブジェクト数を削減する．U_{IP} 中で DP_PSUM2 を適用できるオブジェクトは，(i) 1 つのオブジェクト集合のみに含まれるオブジェクト（図 6 の影付き部の黒点），および (ii) 全てのオブジェクト集合で共有されているオブジェクト（図 6 の斜線部の黒点）である．(i) および (ii) に DP_PSUM2 を適用し，IP 中のオブジェクト集合のローカル MaxRS 確率を計算するアルゴリズムをアルゴリズム 3 に示す．ここで，アルゴリ

Algorithm 3: CalculateLocalMaxRS(IP)

Input: 独立集合 IP

1 $U_\alpha = \{u | \exists R_i, R_j \in IP, u \in (R_i \cap R_j)\}$
2 **for** $\forall R \in IP$ **do**
3 $\quad U_{\hat{R}} = R \setminus (R \cap U_\alpha)$
4 $\quad \Psi_{\hat{R}} =$ DP_PSUM2 $(U_{\hat{R}})$
5 $U_\beta = \{u | u \in \cap_{R \in IP} R\}$
6 $\Psi_\beta =$ DP_PSUM2 (U_β)
7 $U_\gamma = U_\alpha \setminus U_\beta$
8 **for** $\forall \mathcal{W} \in \Omega_{U_\gamma}$ **do** // Ω_{U_γ} は U_γ から生成される全ての `Possible World` の集合
9 $\quad$ **for** $\forall R \in IP$ **do**
10 $\quad\quad$ スコア $f^{\mathcal{W}}(R)$ を計算
11 $\quad\quad \Psi_R^{\mathcal{W}} = \emptyset$
12 $\quad\quad$ **for** $\forall \psi_{\hat{R}} \in \Psi_{\hat{R}}$ **do**
13 $\quad\quad\quad \langle f^{\mathcal{W}}(R) + \psi_{\hat{R}}.score, \psi_{\hat{R}}.prob \rangle$ を $\Psi_R^{\mathcal{W}}$ に挿入
14 $\quad$ **for** $\forall R_i \in IP$ **do**
15 $\quad\quad$ **for** $\forall \psi_i \in \Psi_{R_i}^{\mathcal{W}}$ **do**
16 $\quad\quad\quad prob = \Pr[LMaxRS_q = R_i, S(R) = \psi_i.score | \mathcal{W}]$（式 (5) により計算）
17 $\quad\quad\quad$ **if** $prob \neq 0$ **then**
18 $\quad\quad\quad\quad$ **for** $\forall \psi_\beta \in \Psi_\beta$ **do**
19 $\quad\quad\quad\quad\quad \langle \psi_i.score + \psi_\beta.score, prob \cdot \psi_\beta.prob \cdot \Pr(\mathcal{W}) \rangle$ を $\Psi_{R_i}^{LM.}$ に挿入

Algorithm 4: PROPOSED ALGORITHM

Input: 不確実データ集合 U，長方形の大きさ $q = (a, b)$
Output: P-MaxRS の解 $\mathcal{ANS}$

1 CalculateAR(U, q)
2 DivideARintoIP($\mathcal{AR}$)
3 **for** $\forall IP \in \mathcal{IP}$ **do**
4 $\quad$ **if** $|IP| = 1$ **then** // IP 中の唯一のオブジェクト集合を R_i とする．
5 $\quad\quad \Psi_{R_i}^{LM.} \leftarrow$ DP_PSUM2 (R_i)
6 $\quad$ **else if** $|IP| = 2$ **then**
7 $\quad\quad$ 共通オブジェクト集合および各オブジェクト集合のみに含まれるオブジェクトの集合の 3 つの部分に DP_PSUM2 を適用し，オブジェクト集合のローカル MaxRS 確率を計算
8 $\quad$ **else**
9 $\quad\quad$ CalculateLocalMaxRS(IP)
10 **for** $\forall R_i \in \mathcal{AR}$ **do**
11 $\quad$ 式 (4) により，$\Pr[MaxRS_q = R_i]$ を計算
12 $\quad \mathcal{ANS} = \mathcal{ANS} \cup \langle R_i, \Pr[MaxRS_q = R_i] \rangle$
13 **return** $\mathcal{ANS}$

ズム 3 において，全ての Ψ はスコアと確率のタプルから構成される．

まず，共有オブジェクトの集合 U_α を計算する（1 行）．次に，DP_PSUM2 を用いて，1 つのオブジェクト集合 R のみに含まれるオブジェクトの集合 $U_{\hat{R}}$ の取り得るスコアの和とその確率 $\Psi_{\hat{R}}$ を計算する（2–4 行）．次に，IP 中の全てのオブジェクト集合の共通要素であるオブジェクトの集合 U_β を計算する（5 行）．そして，DP_PSUM2 を用いて，U_β の取り得るスコアの和とその確率 Ψ_β を計算する（6 行）．その後，スコアを計算していないオブジェクト集合 U_γ から生成される各 Possible World $\mathcal{W}$ について考える．まず，$\mathcal{W}$ において，各オブジェクト集合が U_β 以外のオブジェクト集合で取り得るスコアの和と確率をスコア $f^{\mathcal{W}}(R)$ および先に計算した $\Psi_{\hat{R}}$ により計算する（9–13 行）．タプル $\langle score, prob \rangle$ を $\Psi_R^{\mathcal{W}}$ に挿入とは，既に同じスコアをもつタプルが $\Psi_R^{\mathcal{W}}$ に存在する場合は，そのタプルの確率を $\psi.prob = \psi.prob + prob$ とし，同じスコアをもつタプルが存在しない場合は，$\langle score, prob \rangle$ をそのまま $\Psi_R^{\mathcal{W}}$ に挿入する操作である（以降，“Ψ に挿入” はこの操作を示す）．次に，$\Psi_{R_i}^{\mathcal{W}}$ を利用し，$\mathcal{W}$ において，オブジェクト集合 R_i がスコア $\psi_i.score$ のときに独立集合 IP 中で MaxRS である確率を以下の式により計算する（16 行）．

$$\Pr[LMaxRS_q = R_i, S(R) = \psi_i.score | \mathcal{W}] = \psi_i.prob \prod_{R_j \in IP \setminus R_i} \Big(\sum_{\psi_j \in \Psi_{R_j}^{\mathcal{W}}, \psi_j.score < \psi_i.score} \psi_j.prob \Big) \quad (5)$$

この確率が 0 でない場合，$\mathcal{W}$ において，オブジェクト集合 R_i は ψ_i であるときに MaxRS である．よって，U_β および Possible World の確率 $\Pr(\mathcal{W})$ を考慮して計算したローカル MaxRS スコアおよびローカル MaxRS 確率を $\Psi_{R_i}^{LM.}$ に挿入する（18–19 行）．

このアルゴリズムで計算した $\Psi_{R_i}^{LM.}$ は，R_i がローカル MaxRS となるスコアおよびその確率のタプルのリストである．各オブジェクト集合毎の $U_{\hat{R}}$ および U_β が取り得るスコアの和を先に計算しても，Ω_{IP} 中の Possible World を過不足なく計算できる．そのため，アルゴリズム 3 により，オブジェクト集合のローカル MaxRS 確率を正確に計算できる．また，単純手法において Possible World の基となるオブジェクトの数は $|U_{IP}|$ であるのに対して，提案アルゴリズムでは，Possible World の基となるオブジェクトの数を $|U_\gamma|$ に削減している．

以上より，アルゴリズム 1，アルゴリズム 2，アルゴリズム 3，および式 (4) を用いて，各オブジェクト集合の MaxRS 確率を正確に計算できる．P-MaxRS クエリを処理する提案アルゴリズムをアルゴリズム 4 に示す．まず，アルゴリズム 1 およびアルゴリズム 2 により，$\mathcal{AR}$ を計算し，$\mathcal{AR}$ を独立集合に分割する（1–2 行）．次に，独立集合毎に，各オブジェクト集合のローカル MaxRS 確率を計算する（3–9 行）．このとき，独立集合中のオブジェクト集合の数が 1 または 2 の場合，Possible World 毎の計算をせずに各オブジェクト集合のローカル MaxRS 確率を計算できる（4–7 行）．それ以外の場合は，アルゴリズム 3 により，各オブジェクト集合のローカル MaxRS 確率を計算する（8–9 行）．最後に，式 (4) により，各オブジェクト集合 R_i の MaxRS

確率 $\Pr[MaxRS_q = R_i]$ を計算し，$\langle R_i, \Pr[MaxRS_q = R_i]\rangle$ を $\mathcal{ANS}$ に挿入する（10–12 行）．

4. 評価実験

本章では，評価実験の結果を紹介する．提案アルゴリズムの有効性を確認するため，CalculateLocalMaxRS 内で，Possible World の基となるオブジェクト集合を U_{IP} としたもの（BASELINE），および U_α としたもの（BASELINE+）と比較した．評価結果を示すグラフ中では，提案アルゴリズムを PROPOSED と表す．全てのアルゴリズムは C++で実装されており，実験は 3.00GHz Xeon E5-2687W v4 および 512GB RAM で構成される PC 上で行った．

4.1 設　　定

データセット．本実験は，2 種類の実データ（CAR(注1) および NE(注2)）および人工データを用いて行った．CAR および NE は，それぞれカリフォルニア州およびアメリカ北東部の道路ネットワークから取得した空間データであり，実データのオブジェクト数は，それぞれ 2,249,727 および 1,524,453 である．実データを用いた実験では，実データから各オブジェクトの位置を取得し，x 軸方向および y 軸方向の値を，それぞれ [0, 1000000000] の範囲で正規化した．人工データを用いた実験では，各オブジェクトの位置を，$[0, 1000000000]^2$ の範囲の一様乱数により生成した．また，実データおよび人工データにおいて，インスタンスのスコアおよびその生起確率は，以下の方法で人工的に付与した．各オブジェクト毎に，[1,200] の一様分布から決定した整数値を平均とし，分散を 20 とした正規分布を与え，この正規分布に従い [1,200] の整数値となるスコアをもつインスタンスを 2 つ生成した．また，各オブジェクト中のインスタンスの生起確率の合計が 1 となるように，インスタンスの生起確率をランダムに与えた．

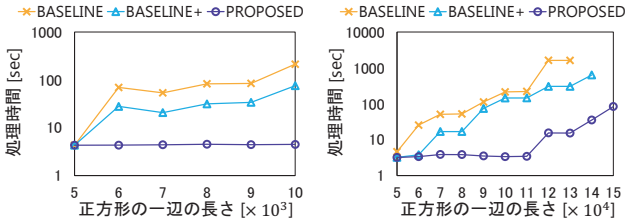
評価環境．本実験では，クエリで指定する長方形を一辺の長さが L である正方形とした．実データによる実験では，各オブジェクトのインスタンスをランダムに変化させたデータセット 50 組を作成し，人工データによる実験では，各オブジェクトの位置およびインスタンスをランダムに変化させたデータセット 50 組を作成した．このように作成したデータセット 50 組に対し，クエリを発行してから P-MaxRS クエリの解 $\mathcal{ANS}$ を求めるまでの処理時間の平均値を調べた．評価結果を示すグラフ中において欠けている点は，開始後 30 時間経過しても終了しなかったものである．

4.2 評 価 結 果

実データによる実験結果．各実データに対して，それぞれ正方形の一辺の長さ L を変化させたときの結果を図 7 に示す．この図において，図 7(a) および図 7(b) はそれぞれ CAR および NE の結果を表す．

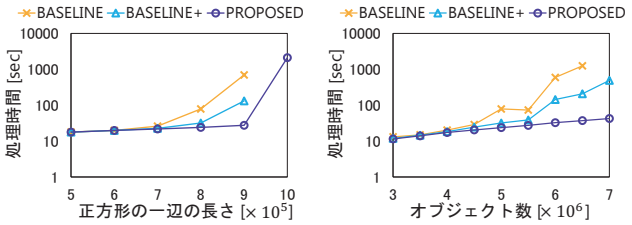
図 7 より，全ての手法において，L が大きくなると処理時間が大きくなることが分かる．L が大きくなると，1 つの正方形

(注1)：http://chorochronos.datastories.org/?q=node/59
(注2)：http://www.dis.uniroma1.it/challenge9/download.shtml



(a) CAR の結果　　(b) NE の結果

図 7　実データにおける正方形の一辺の長さ L の影響



(a) 正方形の一辺の長さ L の影響　　(b) オブジェクト数 $|U|$ の影響

図 8　人工データの評価結果

中のオブジェクトの数が大きくなり，1 つの独立集合に含まれるオブジェクトの数が大きくなる傾向がある．そのため，ローカル MaxRS 確率の計算時に生成する Possible World の数が大きくなる．PROPOSED は，ローカル MaxRS 確率の計算時に生成する Possible World の数を削減しているため，L が大きい場合でも他の手法と比べて処理時間が小さい．

人工データによる実験結果．人工データによる実験結果を図 8 に示す．この図において，図 8(a) は，正方形の一辺の長さ L を変化させた時の結果，図 8(b) は，オブジェクト数 $|U|$ を変化させた時の結果を示す．$|U| = 5000000$ および $L = 800000$ をデフォルト値とし，正方形の一辺の長さの影響を調べる時は，L を [500000, 1000000] の範囲で変化させ，オブジェクト数の影響を調べる時は，$|U|$ を [3000000, 7000000] の範囲で変化させた．

図 7(a) の結果より，L が大きい時，PROPOSED の処理時間が最も小さいことが分かる．BASELINE および BASELINE+ は $L = 1000000$ の時の結果がないが，少なくとも 30 時間経過しているため，BASELINE および BASELINE ＋の処理時間は PROPOSED よりも大きい．これは，L が大きい時，1 つの独立集合に含まれるオブジェクト数が大きいためである．提案手法では，限られた部分に存在するオブジェクトのみから Possible World を生成しているため，この影響が小さい．しかし，$L = 1000000$ の時，PROPOSED の処理時間も大きくなる．これは，U_γ に属するオブジェクト数が大きくなる場合，提案手法の処理時間が増加することを示している．

図 7(b) の結果は，オブジェクト数が大きい時，PROPOSED の処理時間が最も小さいことを示している．一方，BASELINE および BASELINE+の処理時間は非常に大きい．これより，提案手法は計算すべき Possible World の数を削減し，効率的に

各オブジェクト集合の MaxRS 確率を計算できていることが分かる．

5. 関 連 研 究

MaxRS クエリ． MaxRS クエリを効率的に処理する様々なアルゴリズムが提案されている [2], [3], [5], [6], [8], [9], [11], [14]．文献 [9] では，全てのデータをメモリ上に配置できる環境において，文献 [5], [6] では，全てのデータをメモリ上に配置できない環境において，処理時間の削減を目的とした厳密な解を計算するアルゴリズムが提案されている．文献 [14] では，インデックスを構築することにより，さらに処理時間を削減するアルゴリズムが提案されている．また，文献 [11] では，MaxRS クエリの解のスコアの精度を保証した近似解を計算するアルゴリズムが提案されている．文献 [3] では，MaxRS クエリで与えられる長方形の回転を考慮したアルゴリズムが提案されている．文献 [2] では，ストリーム環境において MaxRS を効率的にモニタリングするアルゴリズムが提案されている．文献 [8] では，分散環境であるアドホックネットワークにおいて，トラヒックの削減を目的としたアルゴリズムが提案されている．これらの研究は，データのスコアが確定的であるデータベースを想定しており，これらの研究で提案されているアルゴリズムを不確実データベースに適用することは困難である．

不確実データベース． 不確実データベースの分野において，様々なクエリおよびそれを効率的に処理するアルゴリズムが提案されている [4], [7], [13]．文献 [7] および文献 [4] は，それぞれ不確実データベースにおける Top-k クエリおよび k 最近傍クエリを考えている．文献 [7] は，Top-k となる確率が閾値以上であるデータを取得するクエリを効率的に処理するアルゴリズムを提案している．また，文献 [4] は，k 個のデータ集合がクエリポイントの k 最近傍となる確率が閾値以上であるデータ集合の集合を取得するクエリを効率的に処理するアルゴリズムを提案している．これら以外にも，Top-k 検索および k 最近傍検索に関して，様々なセマンティックのクエリが研究されている．また，文献 [13] では，Bichromatic Reverse Nearest Neighbor の数の期待値が高い k 個のデータポイントを取得するクエリを効率的に処理するアルゴリズムを提案している．これらの研究は，本稿で想定する問題とは異なるため，これらの研究で提案されているアルゴリズムを P-MaxRS クエリ処理に適用することはできない．

6. お わ り に

本稿では，不確実データベースにおける MaxRS クエリを効率的に処理するアルゴリズムを提案した．実環境では，センサデータなどの値が確率的に表されるデータが存在するため，不確実データベースにおけるクエリ処理アルゴリズムを考えることは有用である．提案アルゴリズムは，平面走査法を用いて，P-MaxRS クエリの解となるオブジェクト集合を限定し，部分的に DP_PSUM2 を利用して，計算しなければならない Possible World の数を削減することにより，処理時間を削減する．また，実データおよび人工データを用いた実験から，提案アルゴリズムの有効性を確認した．

提案アルゴリズムは，Possible World の基となるオブジェクト数に対して，指数的に計算時間が増加する．そこで，さらに処理時間を削減するために，精度を保証した近似解を計算するアルゴリズムを設計することを検討している．

謝　　辞

本研究の一部は，文部科学省研究費補助金・基盤研究 (A)(JP26240013) および JST 国際科学技術共同研究推進事業（戦略的国際共同研究プログラム）の研究助成によるものである．ここに記して謝意を表す．

文　　献

[1] Akbarinia, R., Valduriez, P. and Verger, G.: Efficient Evaluation of SUM Queries over Probabilistic Data, *TKDE*, Vol. 25, No. 4, pp. 764–775 (2013).
[2] Amagata, D. and Hara, T.: Monitoring MaxRS in Spatial Data Streams, *EDBT*, pp. 317–328 (2016).
[3] Chen, Z., Liu, Y., Wong, R. C.-W., Xiong, J., Cheng, X. and Chen, P.: Rotating MaxRS queries, *Information Sciences*, Vol. 305, pp. 110–129 (2015).
[4] Cheng, R., Chen, L., Chen, J. and Xie, X.: Evaluating Probability Threshold k-Nearest-Neighbor Queries over Uncertain Data, *EDBT*, pp. 672–683 (2009).
[5] Choi, D.-W., Chung, C.-W. and Tao, Y.: A Scalable Algorithm for Maximizing Range Sum in Spatial Databases, *PVLDB*, Vol. 5, No. 11, pp. 1088–1099 (2012).
[6] Choi, D.-W., Chung, C.-W. and Tao, Y.: Maximizing Range Sum in External Memory, *ACM TODS*, Vol. 39, No. 3, p. 21 (2014).
[7] Hua, M., Pei, J., Zhang, W. and Lin, X.: Ranking Queries on Uncertain Data: A Probabilistic Threshold Approach, *SIGMOD*, pp. 673–686 (2008).
[8] Nakayama, Y., Amagata, D. and Hara, T.: An Efficient Method for Identifying MaxRS Location in Mobile Ad Hoc Networks, *DEXA*, pp. 37–51 (2016).
[9] Nandy, S. C. and Bhattacharya, B. B.: A Unified Algorithm for Finding Maximum and Minimum Object Enclosing Rectangles and Cuboids, *Computers & Mathematics with Applications*, Vol. 29, No. 8, pp. 45–61 (1995).
[10] Qi, J., Zhang, R., Kulik, L., Lin, D. and Xue, Y.: The Min-Dist Location Selection Query, *ICDE*, pp. 366–377 (2012).
[11] Tao, Y., Hu, X., Choi, D.-W. and Chung, C.-W.: Approximate MaxRS in Spatial Databases, *PVLDB*, Vol. 6, No. 13, pp. 1546–1557 (2013).
[12] Wong, R. C.-W., Özsu, M. T., Yu, P. S., Fu, A. W.-C. and Liu, L.: Efficient Method for Maximizing Bichromatic Reverse Nearest Neighbor, *PVLDB*, Vol. 2, No. 1, pp. 1126–1137 (2009).
[13] Zhan, L., Zhang, Y., Zhang, W. and Lin, X.: Finding Top k Most Influential Spatial Facilities over Uncertain Objects, *TKDE*, Vol. 27, No. 12, pp. 3289–3303 (2015).
[14] Zhou, X., Wang, W. and Xu, J.: General Purpose Index-Based Method for Efficient MaxRS Query, *DEXA*, pp. 20–36 (2016).