ガウス分布に対する空間索引のGiSTを用いた実現

兒玉　一樹†　　石川　佳治††,†††

† 名古屋大学大学院情報科学研究科　〒 464–8601 名古屋市千種区不老町
†† 名古屋大学情報基盤センター　〒 464–8601 名古屋市千種区不老町
††† 国立情報学研究所 〒 101–0003 東京都千代田区一ツ橋 2 丁目 1-2
E-mail: †kodama@db.itc.nagoya-u.ac.jp, ††ishikawa@itc.nagoya-u.ac.jp

あらまし　センサや移動履歴などによる移動オブジェクトの位置推定を行う状況では，測定誤差による曖昧性などにより，オブジェクトの位置がしばしば確率分布で表現される．ガウス分布はその中でも最も一般的な確率分布の一つである．本稿では，ガウス分布がオブジェクトとしてデータベース中に蓄積されることを想定し，それらガウス分布に対する確率的範囲検索を定義する．確率的範囲問合せを効率的に処理するための索引手法を提案する．索引は GiST を拡張することで実装する．本稿では，GiST で必要となる関数について，提案する索引手法のアルゴリズムについて述べる．

キーワード　ガウス分布, 空間データベース, 索引構造, GiST，曖昧な位置情報

Implementation of Spatial Indexes for Gaussian Distributions Using GiST

Kazuki KODAMA† and Yoshiharu ISHIKAWA††,†††

† Graduate School of Information Science, Nagoya University　Furo-cho, Chikusa-ku, Nagoya, 464–8601 Japan
†† Information Technology Center, Nagoya University　Furo-cho, Chikusa-ku, Nagoya, 464–8601 Japan
††† National Institute on Informatics 2-1-2, Hitotsubashi, Chiyoda, Tokyo, 101–0003 Japan
E-mail: †kodama@db.itc.nagoya-u.ac.jp, ††ishikawa@itc.nagoya-u.ac.jp

1. まえがき

空間的な情報に基づく空間データベースの問合せの処理技術は，位置情報を扱うアプリケーションにおいて重要となっている．特に，モバイルやセンサネットワークの分野の発展により，これらの分野で空間問合せを処理する技術が必要とされている．たとえば，移動ロボット [2] は通常，センサや移動履歴を用いて自身の位置情報を定期的に推定するが，センサの測定誤差や環境要因から正確な位置情報を得るのは困難であり，曖昧性が生じる．このような曖昧性をもつ各オブジェクトの位置を確率分布を用いて表現する手法が注目されている [10].

本稿では，位置の曖昧性がガウス分布で表現されたオブジェクトに対する空間問合せについて述べる．たとえば，曖昧な位置を持つ移動ロボットが他のオブジェクト (それらも曖昧な位置を持つ) に対して位置に基づく問合せを行う場合などが考えられるため，本稿では，問合せオブジェクト，また，問合せの対象となるオブジェクトがともにガウス分布によって曖昧な位置として表現されている状況を考慮する．確率分布を用いた空間問合せ処理を行う場合，後述のように解の導出過程に数値積分が含まれるのが一般的であるが，数値積分のコストは非常に大きいため，コストを抑えた問合せ処理技術が必要となる．そのアプローチとしては，空間索引を構築し，その索引を用いた問合せ処理を行うのが典型的な手法である．本稿では，ガウス分布による位置推定を行うオブジェクトに対する索引手法を提案し，問合せ処理の効率化を図る．

本稿で提案する索引は，GiST [15] を拡張することによって実現する．GiST では，問合せの種類や格納するキーの型を実装者が定義できる．検索条件や挿入，分割で必要な関数を定義することで B-Tree や R-Tree の振舞いをする索引木や新しいデータ型を扱う索引木の実装が可能である．本手法では，GiST をベースにガウス分布を格納する索引の実装を行う．

以上より，本稿の目的をまとめると以下のとおりである．

- 問合せオブジェクト，その対象となるオブジェクトが**ガウス分布**で表現されている状況における空間データベース問合せを検討
- 空間問合せとして，確率的な概念を考慮した**確率的範囲**

問合せを定義

- 問合せを効率的に処理するため，対象となるガウス分布オブジェクトを格納する索引の構造を提案
- 索引を GiST ベースで実装するために必要な関数を定義

2. 関 連 研 究

ガウス分布に対する索引構造に関して Böhm らが，*Gauss-Tree* と呼ばれる索引構造を [1] で提案している．[1] では，本研究と同様に，問合せオブジェクト，対象となるオブジェクトがともに多次元のガウス分布で表現されている状況を想定している．Gauss-Tree は空間問合せに用いられる R-Tree [3] とは異なり，各オブジェクトのガウス分布のパラメータを各ノードに保持している．根ノードはすべてのデータベースオブジェクトの平均値 μ と誤差 δ の上限値・下限値を格納している．葉ノードでは μ と δ の値が近い値をもつデータベースオブジェクトを格納している．平均や分散の値が近い複数のガウス分布を区分的な近似関数を用いて問合せの処理を行う．類似したパラメータ値を持つガウス分布を同じノードで保持することにより，ガウス分布の近似関数を用いた問合せ処理が可能となる．しかし，[1] では，ガウス分布の各次元が独立であることを前提としている．この制約により，特に多次元の場合では軸に平行なガウス分布しか扱えず，汎用性が高いとは言い難い．

本研究が実装のベースとして用いる GiST は [15] によって提案された索引構造である．[15] では，基本的な性質や挿入，分割等のアルゴリズムを述べ，具体例として B-Tree [4] や R-Tree の実装，集合をデータ型として格納する *RD-Tree* [5] の実装について述べている．GiST は，オープンソースであり，libgist という C++ライブラリが [16] から利用可能である．現在は，PostgreSQL において GiST がサポートされており，SQL 問合せに対する汎用的な索引を構築可能である [19]．

GiST を拡張することによって実現した空間索引が提案されている．[6] では，libgist ライブラリを拡張し，シンプルな空間範囲問合せと最近傍問合せを処理する *M-Tree* を提案している．[7] では，データの領域が時間とともに拡大する場合を考慮した R-Tree の拡張を実装している．また，PostgreSQL の環境で実装した空間索引構造として *SP-GiST* が存在する [17]．いずれの木構造も，対象オブジェクトが確定的な点データで表現されている状況を想定している．本研究では，GiST で確率分布 (ガウス分布) を対象とする索引の実装を図る．

3. 確率的範囲問合せ

本稿では，問合せの対象となるデータオブジェクト（以下**対象オブジェクト**と呼ぶ）と問合せオブジェクトの双方が，曖昧な位置を有しているものとする．それぞれの位置は，**ガウス分布**に基づく確率密度関数により表現されるものとする．一般に，空間の次元数は d 次元であるものとする．

［定義 1］（ガウス分布） d 次元空間において，問合せオブジェクト q の位置が d 次元ベクトルの座標値 $\boldsymbol{x}_q$ を持つ確率が，d 次元ガウス分布の確率密度関数により，

$$p_q(\boldsymbol{x}_q) = \frac{1}{(2\pi)^{\frac{d}{2}}|\boldsymbol{\Sigma}_q|^{\frac{1}{2}}} \exp\left[-\frac{1}{2}(\boldsymbol{x}_q-\boldsymbol{q})^t\boldsymbol{\Sigma}_q^{-1}(\boldsymbol{x}_q-\boldsymbol{q})\right] \quad (1)$$

で表現されるとする．ただし，$\boldsymbol{\Sigma}_q$ は $d\times d$ の共分散行列，$\boldsymbol{q}$ は分布の平均，t はベクトルの転置を表す．同様に，対象オブジェクト o_i $(1 \leqq i \leqq n)$ の位置 $\boldsymbol{x}_i$ を表すガウス分布を

$$p_i(\boldsymbol{x}_i) = \frac{1}{(2\pi)^{\frac{d}{2}}|\boldsymbol{\Sigma}_i|^{\frac{1}{2}}} \exp\left[-\frac{1}{2}(\boldsymbol{x}_i-\boldsymbol{o}_i)^t\boldsymbol{\Sigma}_i^{-1}(\boldsymbol{x}_i-\boldsymbol{o}_i)\right] \quad (2)$$

で表す．$\boldsymbol{\Sigma}_i, \boldsymbol{o}_i$ は上記の定義と同様である． □

これをもとに，確率的範囲問合せを以下のように定義する．

［定義 2］（確率的範囲問合せ） 問合せオブジェクト q と距離の閾値 δ および確率の閾値 θ $(0<\theta<1)$ が与えられたとき，**確率的範囲問合せ**（probabilistic range query, PRQ）を以下のように定義する．

$$PRQ(q,\delta,\theta) = \{o_i | o_i \in \mathcal{O}, \Pr(\|\boldsymbol{x}_q-\boldsymbol{x}_i\| \leqq \delta) \geqq \theta\} \quad (3)$$

ここで $\mathcal{O}$ は対象オブジェクトの集合であり，$\|\boldsymbol{x}_q-\boldsymbol{x}_i\|$ は $\boldsymbol{x}_q$ と $\boldsymbol{x}_i$ のユークリッド距離である．より厳密には，$\Pr(\|\boldsymbol{x}_q-\boldsymbol{x}_i\| \leqq \delta)$ は，

$$\chi_\delta(\boldsymbol{x}_q,\boldsymbol{x}_i) = \begin{cases} 1, & \text{if } \|\boldsymbol{x}_q-\boldsymbol{x}_i\| \leqq \delta \\ 0, & \text{otherwise} \end{cases} \quad (4)$$

という関数を用いて，

$$\Pr(\|\boldsymbol{x}_q-\boldsymbol{x}_i\| \leqq \delta) = \iint \chi_\delta(\boldsymbol{x}_q,\boldsymbol{x}_i)\cdot p_q(\boldsymbol{x}_q)\cdot p_i(\boldsymbol{x}_i) d\boldsymbol{x}_q d\boldsymbol{x}_i \quad (5)$$

と定義できる． □

すなわち，対象オブジェクト o_i が確率的範囲問合せの結果に含まれるためには，それが問合せオブジェクト q から距離 δ 以内にある確率が θ 以上であることが必要である．確率的範囲問合せについては，本グループの過去の研究 [8] ですでに問合せ処理のアプローチを開発しているが，そこでの対象オブジェクトは点オブジェクトであり，問合せオブジェクトのみが曖昧な位置情報を持っていた．本稿のアプローチは，対象オブジェクトも曖昧な位置を持つ確率分布であり，この点で大幅な拡張となっている．

3.1 問合せ条件の判定方式

問合せを処理する素朴なアプローチとしては上記の数値積分を，モンテカルロ法の一種である**重点サンプリング法**（importance sampling）[9] を用いて直接計算することである．指定された確率分布（ここでは 2 つのガウス分布を統合したガウス分布）からランダムにサンプルを抽出し，それが所定の条件を満たす場合に積分結果の値に貢献することになる．

本稿で記す確率的範囲問合せの問合せ条件の評価は以下のように一般化できる．式 (1) および式 (2) で与えられる問合せオブジェクト q および対象オブジェクト o_i の確率密度関数 $p_q(\boldsymbol{x}_q), p_i(\boldsymbol{x}_i)$ をもとに，

$$\boldsymbol{\mu} = \begin{bmatrix} \boldsymbol{q} \\ \boldsymbol{o}_i \end{bmatrix} \quad (6)$$

$$\boldsymbol{\Sigma} = \begin{bmatrix} \boldsymbol{\Sigma}_q & 0 \\ 0 & \boldsymbol{\Sigma}_i \end{bmatrix} \quad (7)$$

という $2d$ 次元の平均ベクトルと $2d \times 2d$ 次元の共分散行列をもとにと定義し，以下のガウス分布を考える．

$$p(\boldsymbol{y}) = \frac{1}{(2\pi)^{\frac{d}{2}}|\boldsymbol{\Sigma}|^{\frac{1}{2}}} \exp\left[-\frac{1}{2}(\boldsymbol{y}-\boldsymbol{\mu})^t\boldsymbol{\Sigma}^{-1}(\boldsymbol{y}-\boldsymbol{\mu})\right] \quad (8)$$

を定義し，式 (8) のガウス分布によりサンプル点 $(\boldsymbol{q}^t, \boldsymbol{o}_i^t)^t$ を求め，距離 $\|\boldsymbol{q}-\boldsymbol{o}_i\|$ が δ 以下である場合のみをカウントの対象とする．

4. 索引の構造

前節で述べたアプローチにより，1 対 1 で各対象オブジェクトに対して問合せ条件を評価すれば，原理的には問合せが実行可能となる．そこで，本節では，各対象オブジェクト集合 $\mathcal{O}$ 上に索引を構築することで，効率的な問合せ処理を実現することを考える．

4.1 上限関数による近似

本稿で想定する対象オブジェクトは任意のガウス分布を持つ確率密度関数 $p_i(\boldsymbol{x}_i)$ であり，その等確率面は楕円体の形状をとる．このような関数は扱いが難しいことから, [8] で我々のグループが提案した，$p(\boldsymbol{x})$ の上限の関数 $p^\top(\boldsymbol{x})$ を用いる．なお，以下では記述を簡単にするため,

$$p(\boldsymbol{x}) = \frac{1}{(2\pi)^{\frac{d}{2}}|\boldsymbol{\Sigma}|^{\frac{1}{2}}} \exp\left[-\frac{1}{2}(\boldsymbol{x}-\boldsymbol{\mu})^t\boldsymbol{\Sigma}^{-1}(\boldsymbol{x}-\boldsymbol{\mu})\right] \quad (9)$$

と定義される，一般的なガウス分布を対象にする．

［定義 3］（上限関数 (Upper-bounding Function)） 共分散行列の逆行列 $\boldsymbol{\Sigma}^{-1}$ のスペクトル分解を

$$\boldsymbol{\Sigma}^{-1} = \sum_{k=1}^{d} \lambda_k \boldsymbol{v}_k \boldsymbol{v}_k^t \quad (10)$$

と表す．ただし，λ_k と $\boldsymbol{v}_k$ はそれぞれ k 番目の固有値と固有ベクトルである．このとき,

$$\lambda^\top = \min\{\lambda_k\} \quad (11)$$

と定義する．共分散行列の固有値はすべて 0 より大きいため，$\lambda^\top > 0$ が成り立つことに注意する．ここで,

$$\mathbf{M}^\top = \lambda^\top \sum_{k=1}^{d} \boldsymbol{v}_k \boldsymbol{v}_k^t = \lambda^\top \mathbf{I} \quad (12)$$

と定義したとき，式 (9) の $\boldsymbol{\Sigma}^{-1}$ を行列 $\mathbf{M}^\top$ で置き換えることで得られる関数を

$$p^\top(\boldsymbol{x}) = \frac{1}{(2\pi)^{\frac{d}{2}}|\boldsymbol{\Sigma}|^{\frac{1}{2}}} \exp\left[-\frac{\lambda^\top}{2}\|\boldsymbol{x}-\boldsymbol{\mu}\|^2\right] \quad (13)$$

と定義する． □

$p^\top(\boldsymbol{x})$ の等確率面は球形となる．ただし，空間全体での積分値が 1 とはならないため，厳密には $p^\top(\boldsymbol{x})$ は確率密度関数ではない．$p^\top(\boldsymbol{x})$ は以下の性質を持つ．

［性質 1］ 任意の $\boldsymbol{x}$ に対して，以下の式が成り立つ [8].

$$p(\boldsymbol{x}) \leqq p^\top(\boldsymbol{x}) \quad (14)$$

この性質を満たし，等確率面が球形の関数のうちで最良のものが $p^\top(\boldsymbol{x})$ である．つまり，$p^\top(\boldsymbol{x})$ は $p(\boldsymbol{x})$ の上限を与えている．図 1 に，同じガウス分布に対する $p(\boldsymbol{x})$ と $p^\top(\boldsymbol{x})$ の等確率面のイメージを示す.

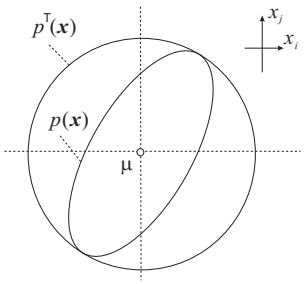

図 1 $p(\boldsymbol{x})$ と $p^\top(\boldsymbol{x})$

4.2 複数のガウス分布の要約情報

本稿で提案する索引手法は，Gauss-Tree にヒントを得て，分布（位置および形状）が似通った対象オブジェクト（すなわちガウス分布）をグループ化することで，R-Tree のような空間索引を構築する．このためには，複数のオブジェクトを索引ノードにまとめたときに，そのノードを代表し問合せ処理をガイドする情報を構築する必要がある．R-Tree の場合は包囲矩形（bounding box）がこれにあたるが，本索引手法では，索引ノード中のオブジェクト（ガウス分布）を代表する要約情報を導出する．

m 個のオブジェクト o_i $(i = 1, 2, \ldots, m)$ を代表する要約関数を以下のように求める．

［定義 4］ **要約関数**

各 o_i $(1 \leq i \leq m)$ に対する上限の関数が,

$$p_i^\top(\boldsymbol{x}) = \frac{1}{(2\pi)^{\frac{d}{2}}|\boldsymbol{\Sigma}_i|^{\frac{1}{2}}} \exp\left[-\frac{\lambda_i^\top}{2}\|\boldsymbol{x}-\boldsymbol{o}_i\|^2\right] \quad (15)$$

と与えられたとする．このとき,

$$\bar{\boldsymbol{o}} = (\bar{o}_1, \ldots, \bar{o}_d)^t = \frac{\sum_{i=1}^m \boldsymbol{o}_i}{m} \quad (16)$$

$$\bar{\lambda}^\top = \frac{\min_{i=1}^m \lambda_i^\top}{2} \quad (17)$$

とし，関数 $cover(\boldsymbol{x})$ を

$$cover(\boldsymbol{x}) = \frac{1}{(2\pi)^{\frac{d}{2}}C} \exp\left[-\frac{\bar{\lambda}^\top}{2}\|\boldsymbol{x}-\bar{\boldsymbol{o}}\|^2\right] \quad (18)$$

と定義する．C は定数であり，定め方は次に述べる． □

4.2.1 定数 C の定め方

C の定め方について述べる．まず,

$$f_i(\boldsymbol{x}) = \frac{cover(\boldsymbol{x})}{p_i^\top(\boldsymbol{x})} \quad (19)$$

$$= \frac{|\boldsymbol{\Sigma}_i|^{\frac{1}{2}}}{C_i} \exp\left[\frac{\lambda_i^\top\|\boldsymbol{x}-\boldsymbol{o}_i\|^2 - \bar{\lambda}^\top\|\boldsymbol{x}-\bar{\boldsymbol{o}}\|^2}{2}\right] \quad (20)$$

と定義する．式 (20) の指数関数中の多項式は下に凸であり $(\lambda_i^\top > \bar{\lambda}^\top$ より)，最小値を持つ．その最小値をとるのは,

$$x_j = \frac{\lambda_i^\top o_{ij} - \bar{\lambda}^\top \bar{o}_{ij}}{\lambda_i^\top - \bar{\lambda}^\top} \quad (j = 1, 2, \ldots, d) \quad (21)$$

においてである．これに基づき，各 i $(i = 1, \ldots, m)$ について $f_i(\boldsymbol{x})$ の最小値を求め，それが 1 となるように C_i の値を調節する．C の値は

$$C = \min_{i=1}^{m} C_i \tag{22}$$

で得られる．このようにすることで，$cover(\boldsymbol{x})$ は m 個の上限の関数に対し，どの $\boldsymbol{x}$ の値についてもそれら以上の値をとる（すなわち，それらをカバーする）指数関数形の関数となる．

4.2.2 要約関数の意義

上記のように定義された要約関数は，$\bar{\boldsymbol{o}}, \bar{\lambda}^\top, C$ という 3 つのパラメータを持つ指数関数形式の関数である．イメージ図を図 2 に示す. [1] では，区分的な関数によりガウス分布の集まりを近似していたのに対し，本提案手法ではより単純な形式の近似を行っている．Gauss-Tree では各次元を独立に扱っていたため，実際には 1 次元の問題であり，より複雑な近似関数を用いることができた．しかし，本稿の場合には，任意形状のガウス分布（実際には上限関数）を統合して検索可能とするため，一般化したより柔軟な近似を行う必要がある．指数関数による近似は最も単純なものであり，上述のように容易に求めることができる．さらに，複数の要約関数を要約した関数を同様の形式で求め，その上位の内部ノードに格納することが可能である．

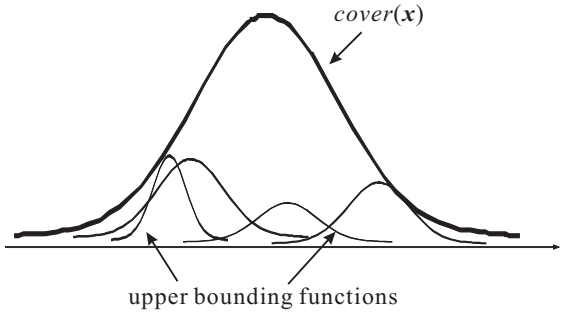

図 2 要約関数のイメージ

4.3 問合せ処理方式

索引の木構造を用いた問合せ処理方式について述べる．式 (1) で示したガウス分布として問合せのガウス分布が与えられ，距離と確率の閾値 δ, θ が与えられたとする．4.1 節で述べた手法により，問合せの分布について上限関数をまず導く．

$$p_q^\top(\boldsymbol{x}_q) = \frac{1}{(2\pi)^{\frac{d}{2}}|\boldsymbol{\Sigma}_q|^{\frac{1}{2}}} \exp\left[-\frac{\lambda_q^\top}{2}\|\boldsymbol{x}_q - \boldsymbol{q}\|^2\right] \tag{23}$$

この問合せ分布の上限関数を，索引部のノード中の索引エントリと比較する．具体的には式 (18) に示した

$$cover(\boldsymbol{x}_c) = \frac{1}{(2\pi)^{\frac{d}{2}}C} \exp\left[-\frac{\bar{\lambda}^\top}{2}\|\boldsymbol{x}_c - \bar{\boldsymbol{o}}\|^2\right] \tag{24}$$

の形式の要約関数が比較対象である．

$cover(\boldsymbol{x}_c)$ 関数は，それがカバーしているいずれの確率密度関数よりも大きい値をとる関数であることから，$\Pr(\|\boldsymbol{x}_q - \boldsymbol{x}_c\| \leqq \delta) < \theta$ が成り立つならば，$cover(x_c)$ がカバーする関数群中には問合せ条件を満たすものはない．逆の場合には，問合せ条件を満たすものが存在しうるため，その部分木の探索が必要となる．式を展開すると，

$$\begin{aligned}
&\Pr(\|\boldsymbol{x}_q - \boldsymbol{x}_c\| \leqq \delta) \\
&= \iint \chi_\delta(\boldsymbol{x}_q, \boldsymbol{x}_c) \cdot p_q^\top(\boldsymbol{x}_q) \cdot cover(\boldsymbol{x}_c) d\boldsymbol{x}_q d\boldsymbol{x}_c && (25) \\
&= \frac{1}{(2\pi)^d|\boldsymbol{\Sigma}_q|^{\frac{1}{2}}C} \iint \chi_\delta(\boldsymbol{x}_q, \boldsymbol{x}_c) \exp[\alpha]\, d\boldsymbol{x}_q d\boldsymbol{x}_c && (26)
\end{aligned}$$

となる．ただし，

$$\alpha = -\frac{\lambda_q^\top}{2}\|\boldsymbol{x}_q - \boldsymbol{q}\|^2 - \frac{\bar{\lambda}^\top}{2}\|\boldsymbol{x}_c - \bar{\boldsymbol{o}}\|^2 \tag{27}$$

である．ここで，上の二重積分の意味を考えると，座標系全体を $\bar{\boldsymbol{o}}$ 方向にシフトしても(注1)積分値は変わらない．すなわち，

$$\begin{aligned}
\alpha &= -\frac{\lambda_q^\top}{2}\|\boldsymbol{x}_q - \boldsymbol{q} + \bar{\boldsymbol{o}}\|^2 - \frac{\bar{\lambda}^\top}{2}\|\boldsymbol{x}_c\|^2 && (28) \\
&= -\frac{\bar{\lambda}^\top}{2}\left(\frac{\lambda_q^\top}{\bar{\lambda}^\top}\|\boldsymbol{x} - \boldsymbol{q} + \bar{\boldsymbol{o}}\|^2 + \|\boldsymbol{x}_c\|^2\right) && (29)
\end{aligned}$$

と変形できるので，

$$\begin{aligned}
&\Pr(\|\boldsymbol{x}_q - \boldsymbol{x}_c\| \leqq \delta) \\
&= \frac{\exp[-\bar{\lambda}^\top/2]}{(2\pi)^d|\boldsymbol{\Sigma}_q|^{\frac{1}{2}}C} \iint \chi_\delta(\boldsymbol{x}_q, \boldsymbol{x}_c) \exp[\beta]\, d\boldsymbol{x}_q d\boldsymbol{x}_c && (30)
\end{aligned}$$

ただし，

$$\beta = \frac{\lambda_q^\top}{\bar{\lambda}^\top}\|\boldsymbol{x}_q - \boldsymbol{q} + \bar{\boldsymbol{o}}\|^2 + \|\boldsymbol{x}_c\|^2 \tag{31}$$

となる．再び二重積分の意味について考えると，この二重積分の部分は以下にのみ依存していることが分かる．

- $\gamma = \lambda_q^\top / \bar{\lambda}^\top$ という比率
- $\eta = \|\boldsymbol{q} - \bar{\boldsymbol{o}}\|$ という距離：ここで考えている分布は等方的であるため，ベクトル $\boldsymbol{q} - \bar{\boldsymbol{o}}$ の方向は重要でない．

つまり，さまざまな (γ, η) のペアに対して事前に二重積分の値 $\iint \chi_\delta(\boldsymbol{x}_q, \boldsymbol{x}_c) \exp[\beta]\, d\boldsymbol{x}_q d\boldsymbol{x}_c$ を計算しておき，表を作っておく．問合せ処理時には，その表を引くことで，二重積分の値を直接求めずに済むことになる．このような表を用いるアプローチは [8] でも活用している．与えられたパラメータのペア (γ, η) にちょうどマッチするエントリがなかった場合は，保守的な（すなわち，false alarm がでないように，積分値を大目に見積もるような）エントリで最も近いものが選ばれる．

以上のアイデアをまとめると，索引木の探索処理は以下のようになる．

（1） まず，$p_q^\top(\boldsymbol{x}_q)$ を求める．

（2） 内部ノードの場合，各エントリ（要約関数）に対し上記のアプローチにより $\Pr(\|\boldsymbol{x}_q - \boldsymbol{x}_c\| \leqq \delta)$ を求める．(γ, η) の表を用いることにより，数値積分は必要としない．確率値が θ 以上である場合には，対応する子ノードをさらに探索することになる．

（3） 葉ノードの場合，各エントリ（ガウス分布）に対応する上限関数 $p_i^\top(\boldsymbol{x}_i)$ と $p_q^\top(\boldsymbol{x}_q)$ について，確率 $\Pr(\|\boldsymbol{x}_q - \boldsymbol{x}_i\| \leqq \delta)$ を評価する．この場合，上の二重積分と同様に，表を用いるアプローチが活用できる．

（4） 評価した確率が θ 以上であるなら，3.1 節で述べた手法により，厳密な確率を求め，それが θ 以上であるならば結果として返す．

(注1)：つまり，$cover(\boldsymbol{x}_c)$ の分布の中心を原点にとりなおす．

5. 索引の実装

本稿では，提案する索引手法を GiST [15] を用いて実装する．本節では，GiST の概要を説明し，提案する索引を適用させるアルゴリズムについて説明する．

5.1 GiST の概要

GiST は高さのバランスが取れた木構造であり，データ型や問合せの種類の拡張が可能である．たとえば，GiST をベースに既存の B$^+$-Tree や R-Tree 等の索引木を実装することが可能である．また，新しいデータ型に特化した索引の実装ができる．

GiST では B+木のような $\langle p, pointer \rangle$ のペアによってキーを与える．p はユーザが定義したクラスによって与えられる**述語 (predicate)** である．たとえば，B$^+$-Tree の場合は述語を整数または文字列として定義し，R-Tree の場合はキーを多次元の点や矩形として定義する．問合せについても同様にクラスを定義することで任意の問合せを実現することが可能である．

GiST を拡張した実装をする場合，一般的に実装者は以下の 4 つの関数を与える必要がある．

- Consistent(E, q)：エントリ $E = (p, ptr)$ の述語 p と，ユーザによる問合せ述語 q が与えられた時，p が q について成立している場合に true，そうでない場合に false を返す
- Union$(E_1, \ldots, E_n)$：データエントリの集合 $S = \{E_1, \ldots, E_n\}$ が与えられた時，その集合内の全エントリに対して true となる述語を返す．この述語を **BP(Bounding Predicate)** [18] と呼ぶ．
- Penalty(E_1, E_2)：新しいデータエントリ E_2 がノード E_1 の部分木に挿入される場合のペナルティ値 (実数) を返す．この値が最も小さいエントリの指すノードに E_2 が挿入される
- PickSplit(N)：ノード N に空きがない時に挿入が発生した場合にそのノードを 2 つに分割する

関数 Consistent は検索時に呼び出される．内部ノードで呼び出された場合，内部ノードのエントリ BP が問合せを満たすかどうかを判定し，Consistent が true であるエントリのみ部分木を探索する．葉ノードの場合，Consistent が true となるエントリが問合せを満たすエントリとなる．関数 Union は，ノード全体の情報を与える BP を求める関数である．たとえば R-Tree の場合，ノード内のエントリの最小包囲矩形 (MBR) を返す関数である．関数 Penalty は挿入時に呼び出され，ペナルティ値が最も小さいノードに新たなノードが挿入される．関数 PickSplit は，ノードにエントリが挿入できなくなった際に呼び出され，適切な分割処理を行う．上記 4 つの関数を用いた検索，挿入，削除，分割等の GiST のアルゴリズムは [15] を参照されたい．

5.2 提案手法のアルゴリズム

本索引手法では，ガウス分布のパラメータを持つエントリを格納する．索引のイメージ図を図 3 に示す．

葉ノードで管理するガウス分布エントリ E は，式 (2) より，平均 $E.\boldsymbol{o}$ および共分散行列 $E.\boldsymbol{\Sigma}$ を保持している．これらのパラメータより，上限関数のパラメータである最小固有値 $E.\lambda^\top$ を求めることができる．内部ノードのエントリ BP は，平均

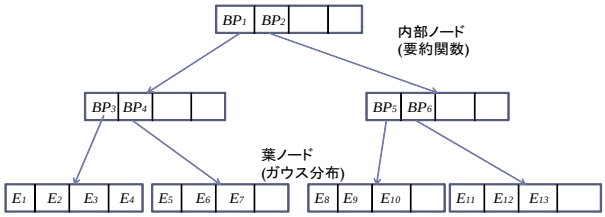

図 3　木構造のイメージ

$BP.\bar{\boldsymbol{o}}$，最小固有値 $BP.\bar{\lambda}^\top$，定数 $BP.C$ を持つ．また，問合せ述語 q は，平均 $q.\boldsymbol{o}$ および共分散行列 $q.\boldsymbol{\Sigma}$ に加え，ユーザが指定する距離および確率の閾値 $q.\delta$, $q.\theta$ を保持する．

5.2.1 Consistent

関数 Consistent は，各エントリが問合せを満たす場合に true を返す関数であるため，本手法においては，各ノードに格納されているガウス分布または要約関数と問合せオブジェクトのガウス分布によって判定する．

4.3 節で述べた問合せ処理アルゴリズムにおける問合せ条件の判定をこの関数で行う．内部ノードの場合，要約関数と問合せオブジェクト (ガウス分布) の上限関数による判定をする．葉ノードの場合は，エントリのガウス分布の上限関数と問合せオブジェクトの上限関数による判定をする．このとき，厳密な積分計算を行わず統計表を用いた確率の見積もりを行う (関数 $Catalog$)．

アルゴリズム 1 Consistent

procedure CONSISTENT(E, q) ▷ E:エントリ q:問合せ
　内部ノードの場合 ▷ $E = (\bar{\boldsymbol{o}}, \bar{\lambda}^\top, C)$(要約関数エントリ)
　$q.\boldsymbol{\Sigma}$ より最小固有値 $q.\lambda^\top$ を求める
　$\gamma = q.\lambda^\top / E.\bar{\lambda}^\top$
　$\eta = \|q.\boldsymbol{o} - E.\bar{\boldsymbol{o}}\|$
　$P = \Pr(\|q.\boldsymbol{x} - E.\bar{\boldsymbol{x}}\| \leqq q.\delta) = Catalog(\gamma, \eta)$ ▷ (γ, η) を元に表から値を見積もる
　if $P \geqq q.\theta$ **then**
　　return true
　else
　　return false
　end if
　葉ノードの場合 ▷ エントリ $E = (\boldsymbol{o}, \boldsymbol{\Sigma})$(ガウス分布エントリ)
　$q.\boldsymbol{\Sigma}$ より最小固有値 $q.\lambda^\top$ を求める
　$\gamma = q.\lambda^\top / E.\lambda^\top$
　$\eta = \|q.\boldsymbol{o} - E.\boldsymbol{o}\|$
　$P = \Pr(\|q.\boldsymbol{x} - E.\boldsymbol{x}\| \leqq q.\delta) = Catalog(\gamma, \eta)$ ▷ (γ, η) を元に表から値を見積もる
　if $P \geqq q.\theta$ **then**
　　return true
　else
　　return false
　end if
end procedure

5.2.2 Union

関数 Union は，ノード内のすべてのエントリをカバーする BP を導く関数である．本索引手法において，BP は複数のガウス分布をカバーする要約関数に相当する．したがって，前節で述べた要約関数の定義に基づいて与えられたエントリから要約関数のエントリを求める．

アルゴリズム 2 Union

1: **procedure** UNION($E_1, \ldots, E_n$) ▷ $E_1, \ldots, E_n$:エントリ
2: 出力は BP(要約関数エントリ)
3: $BP.\bar{\boldsymbol{o}} = avg\{E_1.\boldsymbol{o}, \ldots, E_n.\boldsymbol{o}\}$ ▷ ベクトルの中心点を求める
4: $BP.\bar{\lambda}^{\top} = \frac{min\{E_1.\lambda^{\top}, \ldots, E_n.\lambda^{\top}\}}{2}$ ▷ 事前に共分散行列を用いてガウス分布の最小固有値を求めておく
5: $BP.\boldsymbol{o}, BP.\bar{\lambda}^{\top}$ を用いて $BP.C$ を計算 ▷ 前章の C の求め方を参照
6: **output** BP
7: **end procedure**

5.2.3 Penalty

挿入時のペナルティを求める際，本手法では [13] で提案されている θ 領域に外接する矩形領域を用いる (図 4)．簡単に述べると，まず，一般的なガウス分布である式 (4.1) に対して

$$\int_{(\boldsymbol{x}-\boldsymbol{\mu})^t \boldsymbol{\Sigma}^{-1} (\boldsymbol{x}-\boldsymbol{\mu}) \leqq r_\theta^2} p(\boldsymbol{x}) d\boldsymbol{x} = 1 - 2\theta \tag{32}$$

を満たす r_θ を求める．ガウス分布の中心点から i 番目の次元について大小方向にそれぞれ w_i $(i = 1, 2, \ldots, d)$ の幅を持つとすると，

$$w_i = r_\theta \sigma_i \tag{33}$$

と与えられる．ただし，共分散行列 $\boldsymbol{\Sigma}$ の i 行 i 列の値を $(\boldsymbol{\Sigma})_{ii}$ としたとき，σ_i は

$$\sigma_i = \sqrt{(\boldsymbol{\Sigma})_{ii}} \tag{34}$$

と定義され，i 番目の次元に関する標準偏差に相当する．

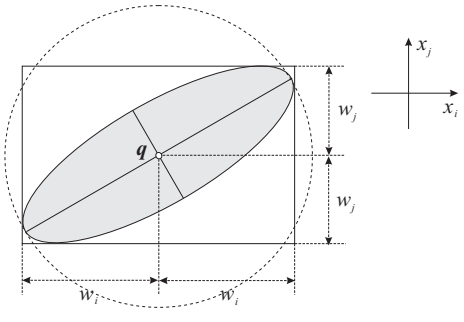

図 4 包囲矩形の利用

この矩形を用いることで，保持しているガウス分布を R-Tree の Bounding Box のように扱うことができる．関数 Penalty は，この手法で作られた矩形を用いて実装する．ノード中の各エントリ (ガウス分布) および挿入されるガウス分布に対して上記の矩形を作成する．エントリ挿入前の最小包囲矩形の面積とエントリ挿入後の最小包囲矩形の面積の差をペナルティ値として返す．この考え方は GiST を用いた R-Tree の実装において利用されている．

アルゴリズム 3 Penalty

1: **procedure** PENALTY(N, E) ▷ N:ノード E:挿入するエントリ
2: ノード N 内の各エントリ $E_1, \ldots, E_n$ に対して θ 領域を包含する矩形領域を求める
3: $Oldvalue = Area(MBR(E_1 \vee, \ldots, \vee E_n))$
4: $Newvalue = Area(MBR(E_1 \vee, \ldots, \vee E_n \vee E))$
5: $pen = Newvalue - Oldvalue$
6: **return** $pen > 0.0$? pen : 0.0
7: **end procedure**

関数 MBR は，矩形で表現された複数のエントリの最小包囲矩形を与える関数である．2 次元の場合，各エントリ $E_1, \ldots, E_n$ の矩形を $(x_{ul1}, y_{ul1}, x_{lr1}, y_{lr1}), \ldots, (x_{uln}, y_{uln}, x_{lrn}, y_{lrn})$ とすると，$(min(x_{ul1}, \ldots, x_{uln}), max(y_{ul1}, \ldots, y_{uln}),$ $max(x_{lr1}, \ldots, x_{lrn}), min(y_{lr1}, \ldots, y_{lrn}))$ を返す関数である．また，関数 $Area$ は引数の領域の大きさを求める関数である．

5.2.4 PickSplit

本索引手法における関数 PickSplit は関数 Penalty 同様に R-Tree の分割法を応用することで実装する．代表的な分割アルゴリズムとして [14] で提案された Quadlatic Split 法のアイデアを活用する．関数 Penalty 同様に，各ガウス分布に対して θ 領域に基づく矩形領域を求め，その矩形を R-Tree の Bounding Box のように扱う．直感的には，MBR が最も大きくなる組合せとなる 2 つの矩形をシードエントリとし，分割後のノードにそれぞれ挿入する (関数 $waste$ による評価)．そして残りのエントリを，挿入前後の MBR の差分が小さくなる方のノードに挿入する (関数 $enlarge$ を用いた判別)．

6. 実験に基づく提案手法の評価

本稿で提案した索引手法の性能を評価する実験について述べる．

6.1 実験環境

1000 × 1000 の領域内に問合せオブジェクトおよび 10,000 個の対象のオブジェクト (ガウス分布) が与えられているものとする．問合せオブジェクトはそれぞれ異なる共分散行列を持つ q_1, q_2, q_3 の 3 種類とし，その各ガウス分布の平均は，領域の中心である (500, 500) で統一した．各問合せオブジェクトの共分散行列は，

$$\boldsymbol{\Sigma}_{q_1} = k \begin{bmatrix} 5 & 0 \\ 0 & 5 \end{bmatrix}$$

$$\boldsymbol{\Sigma}_{q_2} = k \begin{bmatrix} 7 & 2\sqrt{3} \\ 2\sqrt{3} & 3 \end{bmatrix}$$

$$\boldsymbol{\Sigma}_{q_3} = k \begin{bmatrix} 1 & 1.5 \\ 1.5 & 9 \end{bmatrix}$$

とした．オブジェクト q_1 は，等確率面が円である分散を持つ．

アルゴリズム 4 PickSplit

procedure PICKSPLIT(N) ▷ N:ノード
　ノード N 内の各エントリに対して θ 領域を包含する矩形領域を求める
　以下の $waste$ 関数が最大となる，シードエントリ A, $B \in N$ を求める．
　$waste(A,B) = Area(MBR(A \vee B)) - Area(MBR(A)) - Area(MBR(B))$
　LeftNode $\rightarrow \{A\}$
　Rightode $\rightarrow \{B\}$
　残りのエントリを以下の $enlarge$ 関数に基づいてクラスタ X, Y のどちらに割り当てるかを決定する
　$enlarge(X,P) = Area(MBR(X \vee P)) - Area(MBR(X))$
　このとき，$max\{|\ enlarge(X,P) - enlarge(Y,P)\ |\}$ となるエントリ P を選ぶ
　P を $min\{enlarge(X,P), enlarge(Y,P)\}$ に基づいて P をクラスタ X, Y のいずれかに割り当てる
　foreach P on N **do**
　　if $enlarge(X,P) < enlarge(Y,P)$ **then**
　　LeftNode $\rightarrow$ LeftNode $\cup\{P\}$
　　else
　　RightNode $\rightarrow$ RightNode $\cup\{P\}$
　　end if
　end for
　各クラスタを LeftNode, RightNode として分割後のノードとする
end procedure

オブジェクト q_2 は，等確率面が長軸と短軸の比が 3 : 1 で傾き 30 度の楕円である分散を持つ．オブジェクト q_3 は，等確率面がきわめて細い楕円型である分散を持つ．また，k は，分布の曖昧さを示す．解オブジェクトを求める数値積分は，重点サンプリング法を用い，1 回の積分計算に対して 100,000 個の乱数を生成するようにした．このとき，閾値 δ, θ をそれぞれ変化させた場合の

- 索引によって積分が必要と判定したオブジェクト数 (積分候補数)
- 積分によって解であると判定したオブジェクト数 (解数)
- 索引の問合せ処理時間
- 積分計算時間

を評価した．その際，同じ問合せを 10 回行い，その平均値を評価基準とした．実験用プログラムは，GiST の C++ライブラリである libgist [16] を拡張することによって実装した．実験に使用したマシンの CPU は Intel Core 2 Duo E8500 (3.16GHz)，メモリは 4GB，OS は Fedora 12 である．

6.2 実験結果

標準の設定を $\delta = 30, \theta = 0.03, k = 10$ とし，その時の積分候補数，解数，問合せ処理時間，積分計算時間を示す．

積分候補数は 10 回ともすべて同じ値が得られた．いずれの問合せオブジェクトに対しても索引による処理時間は積分計算時間を大きく下回っていることがわかる．問合せ処理時間は，問合せオブジェクト q_1 が優れており，q_3 は若干処理時間が他より多くなっている．これは，索引の内部ノードを探索する時に，$p_q(\boldsymbol{x})$ の代わりにその上限関数 $p_q^{\top}(\boldsymbol{x})$ を用いて候補を絞り込むためだと考えられる．図 1 に示したとおり，同じ確率に対して等確率面を描いたときに，$p_q^{\top}(\boldsymbol{x})$ の等確率面は $p_q(\boldsymbol{x})$ の等確率面に外接する球となる．したがって $p_q(\boldsymbol{x})$ の等確率面の楕円体の形状が球に近い場合には，$p_q^{\top}(\boldsymbol{x})$ の積分値が $p_q(\boldsymbol{x})$ の積分値に近づくため，索引を効率的に探索することが可能であり，逆に楕円体の形状が細い場合には，$p_q^{\top}(\boldsymbol{x})$ の積分値が $p_q(\boldsymbol{x})$ の積分値に比べて大幅に大きくなってしまうため，式 (4.26) の確率が θ 以上になるような探索ノードが増えて処理時間が大きくなるのである．同じ理由から，q_3 の解オブジェクト数の積分候補数に対する割合が他より小さくなっていると考えられる．

表 1　標準の設定 ($\delta = 30, \theta = 0.03, k = 10$) における結果

	積分候補数	解数	問合せ処理時間 [s]	積分計算時間 [s]
q_1	131.0	106.4	0.051	1.82
q_2	152.0	91.9	0.060	1.97
q_3	140.0	67.5	0.065	1.88

表 2　索引の構築時間および木の高さ

索引の構築時間 [s]	木の高さ
0.686	3

索引の構築時間および木の高さを示す．索引の構築時間は，対象オブジェクトのデータに依存するため，前節で調節したパラメータに関係なくほぼ一定の値となった．

6.2.1 パラメータを変化させた場合

表 3　δ を変動させた場合 (問合せオブジェクト q_1)

	積分候補数	解数	問合せ処理時間 [s]	積分計算時間 [s]
$\delta = 10$	47.0	33.7	0.043	0.68
$\delta = 20$	72.0	52.4	0.049	0.96
$\delta = 30$	131.0	106.4	0.051	1.82
$\delta = 40$	225.0	181.5	0.059	2.48
$\delta = 50$	253.0	230.1	0.079	2.77

表 4　δ を変動させた場合 (問合せオブジェクト q_2)

	積分候補数	解数	問合せ処理時間 [s]	積分計算時間 [s]
$\delta = 10$	67.0	27.4	0.050	0.76
$\delta = 20$	101.0	36.8	0.056	1.24
$\delta = 30$	152.0	91.9	0.060	1.97
$\delta = 40$	233.0	141.1	0.068	2.48
$\delta = 50$	313.0	233.3	0.078	3.31

表 5　δ を変動させた場合 (問合せオブジェクト q_3)

	積分候補数	解数	問合せ処理時間 [s]	積分計算時間 [s]
$\delta = 10$	82.0	31.1	0.049	0.94
$\delta = 20$	105.0	49.4	0.054	1.29
$\delta = 30$	140.0	67.5	0.065	1.88
$\delta = 40$	222.0	126.9	0.078	2.37
$\delta = 50$	319.0	238.6	0.091	2.89

表 6　θ を変動させた場合 (問合せオブジェクト q_1)

	積分候補数	解数	問合せ処理時間 [s]	積分計算時間 [s]
$\theta = 0.01$	186.0	144.2	0.079	2.17
$\theta = 0.02$	165.0	122.9	0.064	2.01
$\theta = 0.03$	131.0	106.4	0.051	1.82
$\theta = 0.04$	112.0	92.5	0.049	1.66
$\theta = 0.05$	109.0	84.6	0.046	1.33

表 7　θ を変動させた場合 (問合せオブジェクト q_2)

	積分候補数	解数	問合せ処理時間 [s]	積分計算時間 [s]
$\theta = 0.01$	210.0	155.4	0.087	2.35
$\theta = 0.02$	189.0	141.7	0.076	2.18
$\theta = 0.03$	152.0	92.1	0.060	1.97
$\theta = 0.04$	143.0	81.6	0.053	1.85
$\theta = 0.05$	120.0	70.0	0.051	1.63

表 8　θ を変動させた場合 (問合せオブジェクト q_3)

	積分候補数	解数	問合せ処理時間 [s]	積分計算時間 [s]
$\theta = 0.01$	213.0	124.7	0.103	2.44
$\theta = 0.02$	189.0	92.3	0.089	2.09
$\theta = 0.03$	140.0	67.5	0.065	1.88
$\theta = 0.04$	127.0	58.4	0.059	1.68
$\theta = 0.05$	110.0	43.1	0.051	1.51

ユーザが与えるパラメータである距離の閾値 δ および確率の閾値 θ を変化させたときの結果を示す.

閾値 δ を大きくすると，問合せ条件の範囲が大きくなるため，より多くのノードが探索される．そのため，問合せ処理時間も増加している．同時に，積分の候補数や解オブジェクト数は大きくなる．一方，閾値 θ を大きくすると，当然問合せを満たすオブジェクトは少なくなる．θ が小さい場合，問合せを満たす可能性を持つ内部ノードのエントリが増え，索引を多く探索する必要があるため，問合せ処理コストは大きくなる．しかし，全体を通じて，問合せ処理時間は 0.1[s] を下回るケースが多く，索引の有効性を確認できた.

本稿で提案した索引の構築時間，問合せ処理時間は，積分計算時間を大きく下回っていることがわかった．ただし，本実験では，積分アルゴリズムとしてモンテカルロ法の一種で高速な重点サンプリング法を採用し，サンプル数を 100,000 で固定した．積分計算時間や解オブジェクト数は，積分アルゴリズムや乱数のサンプル数に依存するため，一概には論じられない.

7. ま と め

本稿では，問合せオブジェクトと対象オブジェクトの双方の位置がガウス分布で表現されている状況での空間問合せとして確率的範囲問合せを定義し，効率的に処理を行うために，対象オブジェクト (ガウス分布) を管理する索引構造を提案した．本索引手法は GiST を用いて実装し，評価実験によりその有効性を確認した.

謝　　辞

本研究の一部は，科研費（21013023, 22300034）による.

文　　献

[1] Christian Böhm, Alexey Pryakhin, and Matthias Schubert. “The Gauss-Tree: Efficient Object Identification in Databases of Probabilistic Feature Vectors”. In Proc. ICDE, 2006.
[2] Sebastian Thrun, Wolfram Burgard, and Dieter Fox. “Probabilistic Robotics.” The MIT Press, 2005.
[3] Yannis Manolopoulos, Alexandros Nanopoulos, Apostolos N. Papadopoulos, and Yannis Theodoridis. “R-trees: Theory and Applications”. Springer, 2005.
[4] Douglas Comer. “The Ubiquitous B-Tree”. *Computing Surveys*, 11(2):121-137, June 1979.
[5] Joseph M. Hellerstein and A. Pfeffer. “The RD-tree: An Index Structure for Sets”. Technical Report, University of Wisconsin Computer Science, 1994.
[6] Paolo Ciaccia, Marco Patella and Pavel Zezula. “M-tree: An Efficient Access Method for Similarity Search in Metric Spaces”. Proc VLDB 1997.
[7] Rasa Bliujutk Christian, S. Jensen, Simonas Saltenis and Giedrius Slivinskas. “R-Tree Based Indexing of Now-Relative Bitemporal Data”. Proc VLDB 1998,pp. 345-356.
[8] Yoshiharu Ishikawa, Y. Iijima and J. X. Yu: “Spatial Gange Querying for Gaussian-based Imprecise Query Objects”, ICDE 2009, pp. 676-687 (2009).
[9] W. H. Press, S. A. Teukolsky, W. T. Vetterling and B. P. Flannery: “Numerical Recipies: The Art of Scientific Computing”, Cambridge University Press, 3rd edition (2007).
[10] Jian Pei, Ming Hua, Yufei Tao, and Xuemin Lin.“Query Answering Techniques on Uncertain and Probabilistic Data (Tutorial)”. In Proc. SIGMOD, 2008.
[11] Richard O. Duda, Peter E. Hart, and David G. Stork. “Pattern Classification”. Wiley, 2nd edition, 2000.
[12] Dieter Pfoser and Christian S. Jensen. “Capturing the Uncertainty of Moving-Object Representations”. In Proc. 6th Intl. Symp. on Advances in Spatial Databases (SSD ’99), pp. 111-131, 1999.
[13] 石川 佳治, 飯島 裕一, 「曖昧な位置に基づく空間問合せ処理の効率化」, 電子情報通信学会第 19 回データ工学ワークショップ (DEWS 2008), 2008 年 3 月.
[14] Antonin Guttman. “R-Trees: A Dynamic Index Structure for Spatial Searching.” In Proc. ACM SIGMOD International Conference on Management of Data, pages 47-57, Boston, June 1984.
[15] Joseph M Hellerstein, Jeffery F. Naughton, and Avi Pfeffer. “Generalized Search Trees for Database Systems”. In Proc VLDB 1995
[16] Libgist http://gist.cs.berkeley.edu/
[17] SP-GiST http://www.cs.purdue.edu/spgist/
[18] Megan Thomas, and Joseph M Hellerstein. “Boolean Bounding Predicates for Spatial Access Methods”. Technical Report No. UCB/CSD-02-1174 March 2002
[19] GiST for PostgreSQL http://www.sai.msu.su/ megera/postgres/gist/