空間検索を対象とした
ORDBMSベンチマークテストの開発

島根大学大学院　総合理工学研究科
数理・情報システム学専攻　修士論文
田中研究室
S009322
藤井　宏行

平成 14 年 3 月 20 日

目 次

図 目 次

表 目 次

第1章　はじめに

現在、計算機の世界では数多くのベンチマークが開発されている。我々は、これらのベンチマークテストを実行する事により、計算機に関わるソフトウェア、またはハードウェアの性能を一定の基準で評価・判断ができるようになる。

例えば、現在利用されているベンチマークには、リレーショナル DBMS(RDBMS) のオンライン・トランザクション処理性能を評価する TPC-C、データ意思決定支援の処理性能を評価する TPC-H、TPC-R、HTTP サーバーの GET/POST 処理性能を評価する SPECWeb、WebStone、そして電子商取引をモデルとし、HTTP サーバーと RDBMS の総合的な処理性能を評価する TPC-W などがある。

これら TPC や SPEC などをはじめとした諸団体が策定したベンチマークは、IT 分野の業界標準ベンチマークとして幅広く認知されている。

一方、オブジェクト指向 DBMS(OODBMS) において、マルチメディアデータを取り扱う OODBMS ベンチマークは既に知られている [1][2][3]。しかし、マルチメディアデータの一つである空間データ型を対象としたオブジェクト RDBMS(ORDBMS) ベンチマークは現在あまり広く知られていない。この空間データ型を実装した ORDBMS では、「半径 500m 以内の銀行」などといった空間的な位置関係による検索が可能となる (これを空間検索と呼ぶ)。

本研究では、この空間検索を主な対象とした ORDBMS ベンチマークを新たに開発する事を目標とした。ここで、空間検索を利用したシステム構築例としては、施設管理システム、都市情報システム、顧客管理システム、エリアマーケティング、集配送管理システム、不動産管理システム、防災管理システム、WWW 地図配信システムなどが挙げられる。

このうち本研究で選定した情報システムは、より多くのユーザーに、かつ頻繁に利用され、現在その需要が高いもの、または今後益々その需要が見込まれる情報システムをターゲットとした。具体的には、コンビニやファーストフードの出店計画を想定したエリアマーケティング、モバイル端末と GPS 衛星を利用して取得した現在位置から近隣の位置及び関連情報の取得処理を想定した周辺位置情報システムといった 2 つのタイプの情報システムを選択した。

そこで、本論文ではこれらの 2 つのタイプ (エリアマーケティング、周辺位置情報システム) を対象とした ORDBMS ベンチマークの仕様と実装について述べる。

第2章　OGCが定める空間データ(Geometry仕様)とORDBMSとの関係

2.1　OGCとは

OGC<http://www.opengis.org/>とは、地理情報の相互流通・運用に対する研究、及びそれらに関する標準仕様の策定作業を行っている地理情報システム (GIS) の民間標準化推進機関である。このコンソーシアムは、米国の NASA、NIMA などが有力なスポンサーとなり、世界の有力 GIS ベンダー、DB ベンダー、政府機関、大学などから２００社以上が参加している。

この OGC による地理情報の相互流通・運用のための標準規格には、汎用データベース向けの SQL、Windows 向けの OLE/COM、異機種間接続向けの CORBA といった３つの仕様がある。

2.2　Simple Features Specification for SQL(SFSQL)とは

SFSQL とは、2.1 節で述べた OGC による３つの GIS 世界標準規格のうち、汎用データベース向けの SQL に対して策定された仕様書である。つまりこの仕様書は、GIS の基幹となるジオメトリ、座標変換、データ格納またはデータ取得方法を RDBMS の SQL92 規格に解釈させるため、そのインターフェースや仕様を定めたものである [4]。

この SFSQL の仕様に基づき、SQL92 規格に準拠した ORDBMS に対し、空間データ (Geometry 仕様) の格納、及び検索を可能にした製品群が存在する. 今回本研究で用いた製品は、日立ソフトウェアエンジニアリング社の HiRDB Spatial Search Plug-in である<http://www.hitachi-sk.co.jp/>。

現在では、更に本格的な ORDBMS 規格である SQL99 規格に基づいた製品が IBM、日立などのベンダーにより実装されつつある段階にある [5][6]。

2.3　ORDBMSと空間検索の歴史

ORDBMS と空間検索の歴史に関しては、1986 年に Orenstein が空間検索の概念を提唱してから十数年経ち、現在では図 2.1 と図 2.2 に示す２つの流れが存在している。

尚、本研究では、図 2.1 に示す SFSQL の仕様に基づいて空間検索の機能を実装した ORDBMS(SQL92 規格) 製品、HiRDB Spatial Search を利用している。

1994 年 OGC が設立
↓
1998 年 SFSQL の仕様策定 (SQL92 規格に準拠)
↓
2002 年 空間検索が可能な ORDBMS(SQL92 規格) 製品郡が存在 !!

図 2.1: ORDBMS と空間検索の歴史 1

1986 年 Orenstein(CCA 社) が空間検索の概念を提唱
↓
1993 年 SQL ／ MM プロジェクトの開始
↓
1999 年 SQL ／ MM PART3:Spatial の仕様策定 (SQL99 規格に準拠)
↓
2002 年 空間検索が可能な ORDBMS(SQL99 規格) 製品郡が日立、IBM
オラクルなどの DB ベンダーにより実装中 !!

図 2.2: ORDBMS と空間検索の歴史 2

第3章　空間検索について

地図情報などの空間データ (2次元データ) に対応した ORDBMS では、従来の属性データによる検索の他に、「半径 500m 以内の銀行」などといった空間的な位置関係で検索を行うことができる (一般に空間検索と呼ぶ)。

本研究では、主にこの空間検索の性能評価を行うベンチマークの開発を大きな目標としている。従って、ここでは、まずこの空間検索に関する詳細について述べていく。

3.1　空間データの種類

空間データの種類には、目標物や顧客情報などの「点」データ、道路や線路などの「折れ線」データ、行政界、湖などの「多角形」データなどがある [7]。これらの空間データは、(X,Y) 座標といったベクターデータで表される図形情報である。また、空間データの種別は、OGC の定義ではジオメトリ種別と呼ばれる。

3.2　空間検索の利用例

空間検索を利用したシステム構築例としては、施設管理システム、都市情報管理システム、顧客管理システム、エリアマーケティング、集配送管理システム、不動産管理システム、防災管理システム、Web 地図配信システムなどが挙げられる [7]。

ここでは、上記で挙げたシステムにおける空間検索のイメージを掴むため、「A 駅から半径 500m 以内の銀行」といった具体例を用ることで、空間検索の利用例を説明していく。

3.2.1　埋め込み SQL の作成

空間検索を行う場合、一般的に図 3.1 に示す SQL 文を C や COBOL などのプログラミング言語に埋め込んで利用する事ができる (一般に埋め込み SQL と呼ぶ)。本ベンチマークでは、実際に SQL 文を C 言語に埋め込んで実装している。

```
SELECT フィールド名、…
 FROM テーブル名
  WHERE フィールド名 = 属性検索条件 AND
   WITHIN（フィールド名 、空間検索条件）IS TRUE
```

図 3.1: SQL 文の記述例 1

尚、図 3.1 の WITHIN 関数は、OGC が定める空間データ型に対して利用する関数である。

3.2.2 指定した条件下での空間検索の実行

「A 駅から半径 500 m以内の ATM」といった具体的な空間検索の利用例を図 3.1 に基づいて記述すると図 3.2 のようになる。ここで、この空間検索の対象とするテーブル名を SAMPLE(LOCATION は、空間データ型で定義されたフィールド名) と仮定する。また、SAMPLE テーブルには、あらかじめ表 3.1 に示すような属性データ及び空間データが格納されていると仮定する。

```
SELECT ID、 TYPE、 …
 FROM SAMPLE
  WHERE TYPE = ATM AND
   WITHIN（LOCATION 、 空間検索条件） IS TRUE
```

図 3.2: SQL 文の記述例 2

表 3.1: SAMPLE テーブルの構成

ID	TYPE	…	LOCATION
1	BUS	…	POINT(117、116)
2	CONVENIENCE	…	POINT(299、130)
3	SCHOOL	…	POINT(605、64)
4	RESTAURANT	…	POINT(960、54)
5	STATION	…	POINT(858、146)
6	ATM	…	POINT(171、247)
7	CONVENIENCE	…	POINT(532、239)
8	ATM	…	POINT(715、208)
9	ATM	…	POINT(961、286)
10	HOTEL	…	POINT(286、325
11	RESTAURANT	…	POINT(728、409)
12	ATM	…	POINT(65、559)
13	POLICE	…	POINT(266、605)
14	ATM	…	POINT(481、507)
15	HOTEL	…	POINT(961、754)
16	CONVENIENCE	…	POINT(26、781)
17	HOSPITAL	…	POINT(502、695)
18	RESTAURANT	…	POINT(676、793)
19	ATM	…	POINT(234、897)
20	BUS	…	POINT(481、935)
21	ATM	…	POINT(922、875)

尚、図 3.2 の WITHIN 関数の第一引数には、空間データ型が定義されたフィールド名を指定し、第二引数には、円の半径や中心点 ((X、 Y) 座標) などといった空間検索条件を指定する。ここでは、「A 駅から半径 500 m以内の ATM」といった具体的な例に基づいて、円の半径を 500、中心点を A 駅の位置座標 (858、146) とし、空間データ型のフィールド名を LOCATION として空間検索条件に設定する事になる。

3.2.3　空間検索の視覚的表示例と空間検索後のテーブル抽出結果例

図 3.2 に示す SQL 文に基づく空間検索実行時の視覚的表示例、及びその空間検索実行後の抽出結果例をそれぞれ図 3.3、表 3.2 に示す。ここでは、「A 駅から半径 500 m以内の ATM」といった空間検索を実行することにより、その条件に当てはまる 2 つの銀行が抽出された結果が図 3.3、表 3.2 により、見て取れると思う。

尚、空間データ (LOCATION) の抽出形式には、バイナリ形式とテキスト形式があるが、今回は人間にとって理解しやすいテキスト形式で空間データを抽出した様子を表 3.2 に示している。

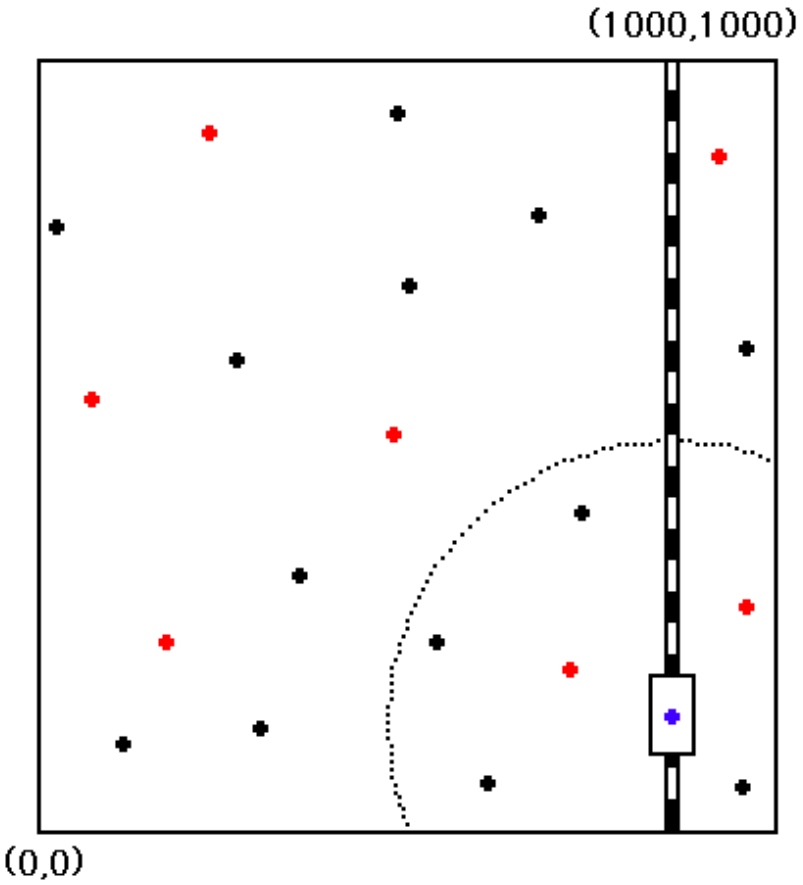

図 3.3: 空間検索の視覚的な表示例

表 3.2: 空間検索後のデーブル抽出結果例

ID	TYPE	···	LOCATION
8	ATM	···	POINT(715、208)
9	ATM	···	POINT(961、286)

第4章　データベース・ベンチマークについて

4.1　ベンチマークとは

ベンチマークとは、与えられた技術環境における特定のアプリケーションの性能を正確に知るために行われるテストのことである。

つまり、ベンチマークは通常、様々なワークロード (負荷) のレベルに対するシステムの反応、あるいはシステムのピークスループットや容量を測定するのに利用される。ベンチマークを利用して何を測定するかは、テスト環境をどのように変化させるか、及びどのような測定データを収集するかによって決定する。

従って、ベンチマークを利用する際には、まず次の２つの問題について考える必要がある [8]。

(1) そのベンチマークは、具体的に何を測定するものなのか？
(2) そのベンチマークで使われる環境条件は、測定の対象となる構成が処理する通常のワークロードとどの程度似ているか？

例えば、TPC-C ベンチマークは、RDBMS のオンライン・トランザクション処理性能を測定するように設計されている。しかし、オンライン・トランザクション処理は、銀行などにおいて頻繁に利用されるが、その他のビジネス情報の処理では必要ない事も多い。従って、オンライン・トランザクション処理を頻繁に利用しないユーザーが TPC-C ベンチマークの結果に頼って RDBMS 用のハードウェアやソフトウェアを選択したとしたら、上記で挙げた２つの問題を考慮していないことになり、誤った選択となる。

今回の本研究の目的は、空間検索を対象とした ORDBMS ベンチマークを開発することである。上記のことを踏まえると、何を測定するにせよ、実際のワークロードをうまく表現できているかどうかが、そのベンチマークの良し悪しを大きく左右すると言える。

4.2　標準ベンチマークを策定する団体

最も一般的なベンチマークプログラム、あるいは最も広く知られているベンチマーク結果は、計算機のハードウェアベンダーとソフトウェアベンダーの協力によって作られた団体が発表しているものである。例えば、RISC ワークステーションを製造している企業は、SPEC(Standard Performance Evaluation Corporation)<http://www.specbench.org/>のスポンサーとなっている。一方、DBMS を販売している企業は他の利益団体と共同で TPC(Transaction Processing Performance Council)<http://www.tpc.org/>を運営している [8]。

特に、TPC ベンチマークは、データベースサーバーを利用したトランザクション処理性能の評価に重点を置いており、IT 関係者の間では、DBMS のトランザクション処理性能を対象とした標準ベンチマークとして、現在幅広く認知されている。

4.3 標準的なベンチマークの種類

現時点 (2002 年 2 月) までに公表されているデータベース、及び HTTP サーバーに対する標準的なベンチマークの種類を表 4.1 に示す (標準的なベンチマークという言葉の意味として、その仕様やソースコードなどといった実装法が Web 上、または論文として公開されているベンチマークをここでは指す)。ここで、注目すべき点は赤色で示している ORDBMS に対する標準的なベンチマークが現在あまり広く知られていないということである。

従って、本研究ではこの ORDBMS に対するベンチマークを新たに開発することを目標とした。

表 4.1: 標準的なベンチマークの種類

	RDBMS	OODBMS	ORDBMS	HTTP Server	測定ターゲット
Wisconsin	○	×	×	×	基本的な関係問い合わせ
TPC-C	○	×	×	×	オンライン・トランザクション
TPC-H	○	×	×	×	DSS(Ad Hoc Queries)*
TPC-R	○	×	×	×	DSS(Business Report)*
WebStone	×	×	×	○	HTTP サーバー
SPECWeb	×	×	×	○	HTTP サーバー
TPC-W	○	×	×	○	電子商取引
001[1]	○	○	×	×	エンジニアリング・サーバー

*ここで DSS は、Decision Support System の略であり、意思決定支援システムのことを意味している。

4.4 今回開発した本ベンチマークについて

ここからは、具体的に本ベンチマークの仕様について順次述べていく。まず本ベンチマークでは、空間検索を主な対象とし、かつ現実世界において実際に利用されているワークロード・モデルを選定している。具体的に本ベンチマークでは、エリアマーケティングと周辺位置情報システムといった 2 つのタイプのワークロード・モデルを選定した。これら 2 つのタイプは、それぞれビジネス分野のユーザー、そして我々を含めた一般ユーザーにとって現在需要があり、かつ今後益々その需要が見込まれる空間検索を利用した情報システムであると思われるため選定した。

ここからは、これら 2 つのタイプのワークロードの表現に関する詳細を、それぞれ 4.4.1 節、4.4.2 節で述べる。

4.4.1 本ベンチマークのワークロード の表現１ (TYPE-A:エリアマーケティングの場合)

本ベンチマーク Type-A のワークロードは、コンビニやファーストフードなどの出店計画 (エリアマーケティング) を行う場合を想定している。この場合、「ある出店予定計画場所 (X、Y) から半径 500 m以内にある顧客レコードを空間データが定義・格納されたフィールドを利用して検索し、更に属性データから総世帯数、総人口数、及び消費支出の平均などを集計・抽出する」などといった一連の処理が考えられる。

このような空間検索を伴う一連の処理は、ビジネス分野において頻繁に利用されるデータベース・トラ

ンザクション処理の一つと考えられる。従って、本ベンチマークテストでは、上記に相当するデータベース・トランザクション処理を TYPE-A のワークロードとして採用した。

ワークロード TYPE-A の具体的な構成は、5.3 節を参照のこと。

4.4.2 本ベンチマークのワークロードの表現２ (TYPE-B:周辺位置情報システムの場合)

本ベンチマーク Type-B のワークロードは、モバイル端末と GPS 衛星を利用して取得した現在位置から近隣の位置及び関連情報の取得処理 (周辺位置情報検索) を行う場合を想定している。

この場合、「PHS や GPS 携帯といったモバイル端末と GPS 衛星を利用して取得した現在位置から半径 500m 以内の最寄り (最も近い) 駅、及び ATM、バス停、コンビニ、レストラン、警察、ホテル、病院、学校などを空間データ型が定義・格納されたフィールドを利用して検索し、更にそれらに関連する情報を抽出する (例えば、最寄り駅までの最短距離や時刻表を抽出する)」などといった一連の処理が考えられる。

このような空間検索を伴う一連の処理は、我々をはじめとした一般ユーザーが今後頻繁に利用するだろうデータベース・トランザクション処理の一つと考えられる。従って、本ベンチマークでは、上記に相当するデータベース・トランザクション処理を TYPE-B のワークロードとして採用した。

ワークロード TYPE-B の具体的な構成は、6.3 節を参照のこと。

4.5 空間データの生成条件

ワークロードの表現においては、4.4.1 節や 4.4.2 節で示したワークロードの表現の他に、ワークロードのデータ構造を定義する必要がある。ワークロードで処理されるサンプルデータが、より現実世界に近い生成関数モデルで生成されているかどうかが本ワークロードの表現の良し悪しに大きく左右するからである。

ここでは、本ベンチマークのワークロードの構成において特に影響が大きいと思われる空間データ (例：顧客や建物の位置データなど) の生成条件について述べる。

本ベンチマークでは、空間データを一様分布としてまず生成した。しかし、現実世界において空間データが一様分布であることは稀であると思われるため、空間データを二次元正規分布に基づき生成する事で、より現実世界に近い疎と密な部分が混在する空間データも用意した。

そして、本ベンチマークテストでは、TYPE-A のエリアマーケティングに対して、空間データの生成条件の違いにより、表 4.2 に示すように TYPE-A-1、TYPE-A-2 という２種類のワークロードを定義した。一方、TYPE-B の周辺位置情報システムに対しては、表 4.2 に示すように TYPE-B という１種類のみのワークロードを定義した。

属性データも含めたサンプルデータの生成条件に関する詳細は、5.2 節、6.2 節を参照のこと。

表 4.2: ３種類のワークロードに対する空間データの生成条件

ワークロード名	空間データの生成条件
TYPE-A-1,TYPE-B	一様分布
TYPE-A-2	二次元正規分布

第5章　本ベンチマークテストの仕様と実装１ (TYPE-A:エリアマーケティングの場合)

5.1　データベース・テーブルの構成

本ベンチマークテスト TYPE-A(エリアマーケティング) で使用されるデータベース・テーブルは、表 5.1 に示す CUSTOMER テーブルから構成されている。

●**CUSTOMER テーブル**：顧客に関するデータを格納したテーブルである。顧客の ID、名前、住所、電話番号、郵便番号、データ登録日、世帯人数、世帯消費支出といった属性データの他に、顧客の位置情報として空間データが含まれる。

また、このテーブルは、空間データ (LOCATION フィールド) を利用して、空間検索が可能である。

表 5.1: CUSTOMER テーブルの構成

フィールド 名	フィールド の定義	意味
ID	INTEGER	顧客 ID
NAME	VARCHAR(30)	顧客の名前
ADDRESS	VARCHAR(50)	顧客の住所
PHONE	VARCHAR(12)	顧客の電話番号
ZIP	CHAR(7)	顧客の郵便番号
SINCE	DATE	データ登録日
NUMBER	SMALLINT	顧客の世帯人数
EXPENDITURE	SMALLINT	顧客の世帯消費支出
LOCATION	GEOMETRY	顧客位置データ

尚、本ベンチマークテストでは、CUSTOMER テーブルが一つだけ存在し、5.2 節に基づいて生成されたサンプルデータが９万レコード 格納されている。

5.2　データベース格納サンプルデータの生成条件

5.2.1　属性データの生成条件 (TYPE-A-1, TYPE-A-2 共通)

ここでは、CUSTOMER テーブルの ID、NAME、ADDRESS、PHONE、ZIP、SINCE、NUMBER、EXPENDITURE フィールド に格納する属性データの生成条件とデータ制約について表 5.2 に示す。

ここで、NUMBER と EXPENDITURE の値は、表 5.2 に示す平均と図 5.1 に示す分散共分散行列 Σ を利用し、図 5.1 の式 (1) に基づいて、多変量正規分布として生成している。

また、下記の生成条件に基づいて生成されたNUMBERとEXPENDITUREの値は、必ず１以上の整数とし、多変量正規分布の平均、分散、及び分散共分散行列Σの値は、総務省統計局<http://www.stat.go.jp/>の家計調査データ (H13年4〜6月期) を基に推定した。

表 5.2: 属性データの生成条件とデータ制約

列名	生成条件とデータ制約
ID	一意制約 (1、2、…、90000)
NAME	最大長30のランダムな英文字列
ADDRESS	最大長50字以内のランダムな英数字列
PHONE	一意制約 (0852000001、…、0852090000)
ZIP	長さ７のランダムな数字列
SINCE	範囲 (1990-01-01〜1999-12-31) の日時型
NUMBER	多変量正規分布 (平均2.67、分散はΣを参照)
EXPENDITURE	多変量正規分布 (平均26.64、分散はΣを参照)

$$\boldsymbol{y_i} = C\boldsymbol{z_i} + \boldsymbol{\mu} \quad (\Sigma = CC',\ i = 1、2、\cdots、90000) \qquad (1)$$

$\boldsymbol{y_i}$：式 (1) により生成される多変量正規乱数ベクトル

$\boldsymbol{z_i}$：平均0、分散 $(1)^2$ の単変量正規乱数ベクトル

$\boldsymbol{\mu}$：平均ベクトル

$$\Sigma = \begin{pmatrix} 1.78 & 7.76 \\ 7.76 & 42.23 \end{pmatrix}$$

図 5.1: 多変量正規分布の算出式

5.2.2 空間データの生成条件１ (一様分布の場合)

TYPE-A-1(一様分布) の場合におけるCUSTOMERテーブルのLOCATIONフィールドに格納する空間データの生成条件とデータ制約について表5.3に示す。

表 5.3: TYPE-A-1 に対する空間データの生成条件

列名	生成条件とデータ制約
LOCATION	一様分布 ((1、1)〜(10000、10000))

表5.3をふまえ、TYPE-A-1の生成条件に基づいて生成された空間データの視覚的表示を図5.3に示す。そして、図5.3のデータに対し、円による空間検索を実行した時の視覚的表示を図5.4に示す。

5.2.3 空間データの生成条件２ (二次元正規分布の場合)

TYPE-A-2(二次元正規分布) の場合における CUSTOMER テーブルの LOCATION フィールドに格納する空間データの生成条件とデータ制約について表 5.4 に示す。

表 5.4: TYPE-A-2 に対する空間データの生成条件

列名	生成条件とデータ制約
LOCATION	二次元正規分布 ((1、1)〜(10000、10000))

空間データ (LOCATION) のＸ、Ｙ座標の値は、図 5.2 中の式 (2)、(3) に基づいて、二次元正規分布として生成する。

ここで、二次正規分布を 9 つ生成する事により顧客位置の疎と密が混在する空間データを作成している。この時、生成する 9 つの二次元正規分布に対する中心座標、及び半径の分散 (図 5.2 における σ^2) の値は、表 5.5 に示す値からそれぞれユニークかつランダムに選択している。

尚、図 5.1、5.2 と表 5.4、5.5 をふまえ、TYPE-A-2 の生成条件に基づいて生成された空間データの視覚的表示を図 5.5 に示す。

Ｘ座標 $= |r| \cos\theta$ (2)
Ｙ座標 $= |r| \sin\theta$ (3)
θ: 一様分布 $(0 \leq \theta \leq 2\pi)$
r：正規分布 (平均 0、分散σ^2)

図 5.2: 空間データ ((X、Y) 座標) の算出式

表 5.5: 二次元正規分布に対する中心座標と半径の分散

中心座標	(2000、2000)、 (5000、2000)、 (8000、2000) (2000、5000)、 (5000、5000)、 (8000、5000) (2000、8000)、 (5000、8000)、 (8000、8000)
分散 σ^2	$(300)^2$、 $(350)^2$、 $(400)^2$、 $(450)^2$、 $(500)^2$ $(550)^2$、 $(600)^2$、 $(650)^2$、 $(700)^2$

5.2.4 生成した空間データの視覚的表示

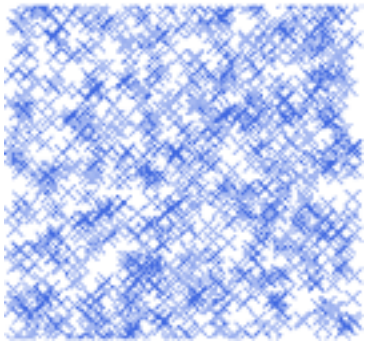

図 5.3: 一様分布に基づいて生成した空間データの表示 (1000 × 1000)

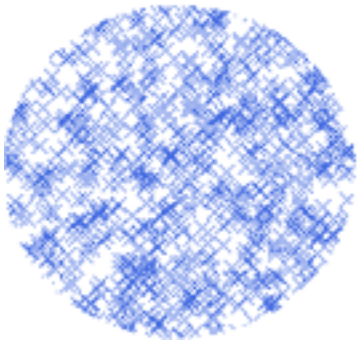

図 5.4: 円による空間検索実行時の空間データの表示例 (円の半径が 500 の時)

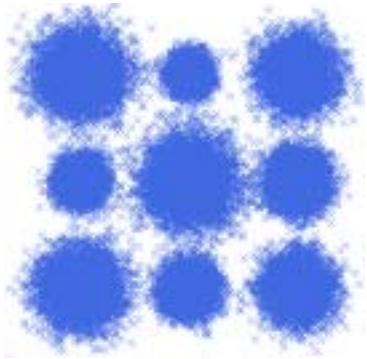

図 5.5: 二次元正規分布に基づいて生成した空間データの表示 (10000 × 10000)

5.3 ワークロードの構成

ここでは、ワークロード TYPE-A のフローチャート、及びワークロードの定義をそれぞれ 5.3.1 節、5.3.2 節で述べる。尚、ワークロードの実装には、埋め込み C を利用している。

5.3.1 フローチャート

ワークロード TYPE-A のフローチャートを図 5.6 に示す。

(1)Generate Spatial Condition Routine の実行：(C 言語で記述)
「空間検索条件となる円の半径と中心座標 (X, Y) の生成」
↓
(2)Spatial Search and Data Extraction Query の実行：(SQL で記述)
「CUSTOMER テーブルに対し (1) のルーチンで生成された空間検索条件で空間検索を実行し、更に、総世帯数、総人口数、平均消費支出を集計・抽出」

図 5.6: ワークロード TYPE-A のフローチャート

5.3.2 ワークロードの定義

●Generate Spatial Condition Routine：

図 5.6 の (1) に示すこのルーチンは、表 5.6 に示す生成条件に基づいて円の半径と中心座標といった空間検索条件を生成している。このルーチンは、C 言語により実装している。

尚、このとき、(1) のルーチンで生成される円は必ず (1、1)〜(10000、10000) の範囲内に収まる (ちなみに、本ベンチマークの空間データは、(1,1)〜(10000,10000) の範囲内で生成している)。

表 5.6: 空間検索条件の生成条件 1

	生成条件
円の中心座標	一様分布 ((1001,1001)〜(9000,9000))
円の半径	正規分布 (平均 500、分散 $(30)^2$)

●Spatial Search and Data Extraction Query：

図 5.6 の (2) に示すこのクエリーは、(1) のルーチンにより生成された空間検索条件を利用して、空間検索を実行するクエリーである。このクエリーは、図 5.7 に示す SQL 文により実装した。

図 5.7 において、(1) のルーチンにより生成された空間検索条件は、埋め込み変数 condition を利用して C 言語から引き渡され、逆に SQL 文による総世帯数、総人口数、平均消費支出といったデータの集計・抽出結果は、それぞれ埋め込み変数 counter、sum_number、ave_expenditure に引き渡される。

尚、図 5.7 の WITHIN 関数は、OGC が定める空間データ型に対して利用する関数である。

```
EXEC SQL
 SELECT COUNT(*), SUM(NUMBER), AVG(EXPENDITURE)
        INTO  counter, sum_number, ave_expenditure
  FROM CUSTOMER
   WHERE WITHIN(LOCATION,condition) IS TRUE;
```

図 5.7: SQL 文による Spatial Search and Data Extraction Query(TYPE-A) の定義

5.4 ベンチマーク測定プログラムの構成

ここでは、本ベンチマーク TYPE-A の測定プログラムのフローチャート、ループ条件、及び測定結果について、それぞれ 5.4.1 節、5.4.2 節、5.4.3 節で述べる。

5.4.1 フローチャート

ベンチマーク TYPE-A の測定プログラムのフローチャートを図 5.8 に示す。図 5.8 に示してある【ワークロードの定義】について、TYPE-A では、具体的に 5.3.1 節に示す流れでルーチンやクエリーを実行する。

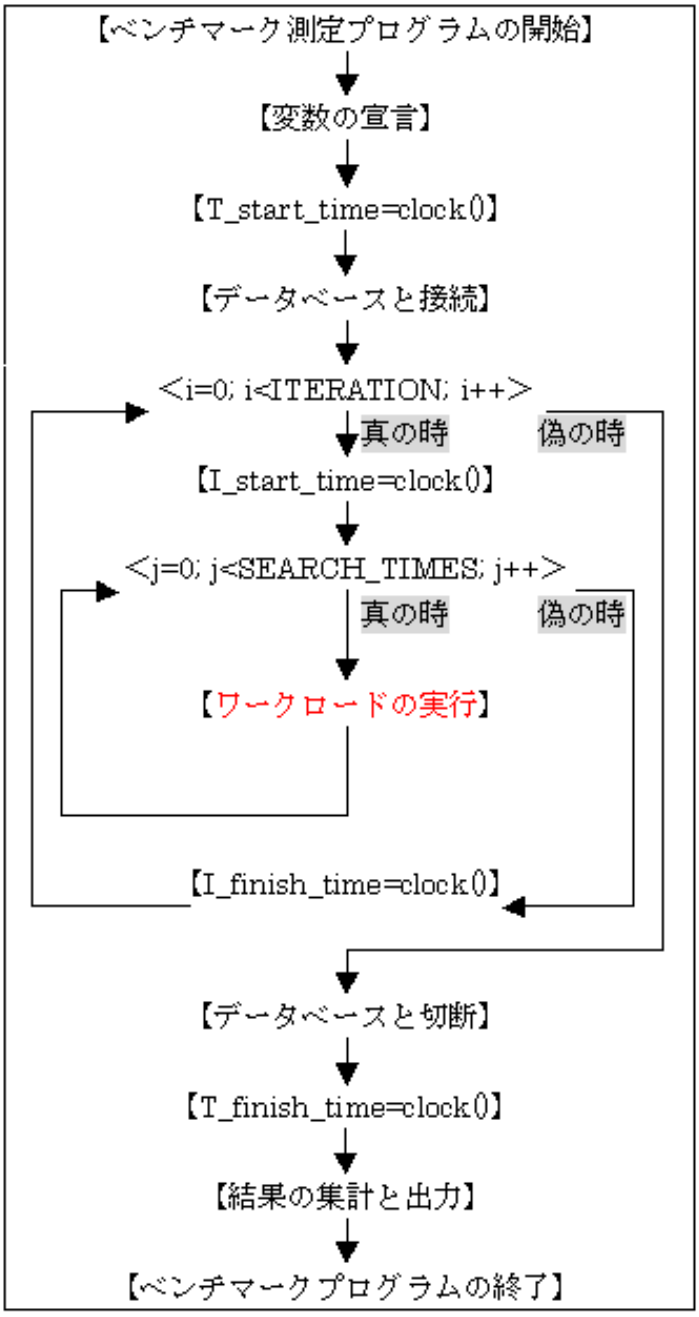

図 5.8: ベンチマーク測定プログラムのフローチャート

5.4.2 ループ条件

図 5.8 に示す本ベンチマークの測定プログラムのループ条件について、表 5.7 に示す。

表 5.7: 測定プログラムのループ条件

ループ名	意味
SEARCH_TIMES	1回の ITERATION(ベンチマークテスト) に対して、ワークロードを処理する回数
ITERATION	1回のベンチマーク測定プログラムの実行に対して、ベンチマークのテスト結果が測定される回数

尚、ベンチマークテストでは、ITERATION を 10、SEARCH_TIMES を 100 に設定している。

5.4.3 測定結果について

本ベンチマークでは、次に示すレスポンスタイムを測定結果として採用している。

●ベンチマークテストを1回実行した時のレスポンスタイム：SEARCH_TIMES × (ワークロード 1 回の処理時間)

尚、本ベンチマークでは、I_start_time と I_finish_time を用いて計測している。

●総レスポンスタイム：$\sum_{i=1}^{10}$ (ベンチマークテスト i 回目のレスポンスタイム) + (データベースとの接続/切断時間)

尚、本ベンチマークでは、T_start_time と T_finish_time を用いて計測している。

5.5 テーブルの定義

```
CREATE TABLE CUSTOMER
(
ID            INTEGER      NOT NULL,
NAME          VARCHAR(30)  NOT NULL,
ADDRESS       VARCHAR(50)  NOT NULL,
PHONE         VARCHAR(12)  NOT NULL,
ZIP           CHAR(7)      NOT NULL,
SINCE         DATE         NOT NULL,
NUMBER        SMALLINT     NOT NULL,
EXPENDITURE   SMALLINT     NOT NULL,
LOCATION      GEOMETRY
)
PRIMARY KEY (ID);
```

図 5.9: SQL 文による CUSTOMER テーブルの定義

CUSTOMER テーブルは、図 5.9 に示す SQL 文により定義した (図 5.9 の GEOMETRY 型が OGC の定める空間データ型である)。

また、ID フィールドは主キーとして定義し、空間データ型で定義した LOCATION フィールドには、空間インデックスを定義した (5.6 節参照)。

5.6 空間インデックスの定義

本ベンチマークでは、図 5.9 で定義した CUSTOMER テーブルの LOCATION フィールドに対して、主キーである ID フィールドとは別のインデックスを図 5.10 に示す SQL 文により定義した。また、LOCATION フィールドに対するインデックス名は、TypeA_Spatial とした。

尚、空間データ型である LOCATION フィールドに対するインデックスは、空間インデックスと呼ばれ、この空間インデックスに対する定義は、ユーザーが利用する DBMS に依存する。ここでは、本研究で利用している HiRDB Spatial Search を前提として説明している。

```
CREATE INDEX TypeA_Spatial USING TYPE SPATIAL ON
CUSTOMER(LOCATION) PLUGIN 'EXTENT=1,1,10000,10000;
DEPTH=6,PRECISION=1';
```

図 5.10: 空間インデックスの定義 1

ここで、図 5.10 に示す LOCATION フィールドに対する空間インデックスの定義のうち、DEPTH=6 について簡単に説明する. まず、HiRDB Spatial Search では、空間データを 4 分木を利用して検索する。この時、インデックスを定義する領域の範囲 (EXTENT) と深さレベル (DEPTH) を指定する必要がある。

深さレベルとは、図 5.10 の EXTENT で指定した領域範囲 (ここでは、(1、1)〜(10000、10000)) を順次 4 つのセルに分割していった時の木の高さである。レベルが深くなれば最小セルのサイズが小さくなっていき、その最小セルに含まれる図形数 (空間データ) の平均が 100〜800 程度 (整数座標系の場合) である時、検索性能を向上できる [7]。

このことをふまえ本ベンチマークでは、最適な深さレベルを算出し、深さレベル (DEPTH) を 6 に設定した。

5.7 実行環境

本ベンチマークテス TYPE-A の実行環境を下記の表 5.8 に示す。

表 5.8: 本ベンチマークテストの実行環境

OS	Windows2000 Advanced Server
DBMS	HiRDB Single Server Ver6.0
CPU	Pentium　866MHz
メモリ	256MB
キャッシュ	256KB
ソフトウェア	Spatial Search Plug-in Ver3.0 Object Option Ver6.0

第6章　本ベンチマークテストの仕様と実装２ (TYPE-B:周辺位置情報システムの場合)

6.1　データベース・テーブルの構成

本ベンチマークテスト TYPE-B(周辺位置情報システム) で使用されるデータベース・テーブルは、表 6.1、表 6.2 に示す MAIN、BUS といった２種類のテーブルから構成されている。

●**MAIN テーブル**：目標物に関するデータを格納したテーブルである。目標物の ID、タイプ、名前、住所、電話番号、郵便番号、データ登録日、目標物の関連情報が定義されたテーブル名といった属性データの他に、目標物の位置情報として空間データが含まれる。

また、このテーブルは、空間データ (LOCATION フィールド) を利用して、空間検索が可能である。

表 6.1: MAIN テーブルの構成

フィールド名	フィールドの定義	意味
ID	INTEGER	目標物 ID
TYPE	VARCHAR(10)	目標物のタイプ
NAME	VARCHAR(30)	目標物の名前
ADDRESS	VARCHAR(50)	目標物の住所
PHONE	VARCHAR(12)	目標物の電話番号
ZIP	CHAR(7)	目標物の郵便番号
SINCE	DATE	データ登録日
REF_TABLE_NAME	VARCHAR(12)	目標物の関連情報が定義されたテーブル名
LOCATION	GEOMETRY	目標物の位置データ

尚、本ベンチマークテストでは、MAIN テーブルが一つだけ存在し、6.2 節に基づいて生成されたサンプルデータが３万レコード格納されている。

●**BUS ××テーブル**：バス停の時刻表に関するデータを格納したテーブルである。××にはユニークかつランダムな文字列を指定する (例：BUS1、BUS2 など)。このテーブルには、バス到着時刻 (時)、バス到着時刻 (分) といった属性データが含まれる。ここで、バス到着時刻 (分) は SQL99 の新しいデータ構造である Array 型で定義した。

尚、本ベンチマークテストでは、ユニークなテーブル名の BUS テーブルが１００テーブル (BUS1、BUS2、…、BUS100) 存在し、6.2 節に基づいて生成されたサンプルデータがそれぞれの BUS テーブルに対して、２１レコード (4 時〜24 時の間のバス到着時刻を想定) づつ格納されている。

表 6.2: BUS ××テーブルの構成

フィールド名	フィールドの定義	意味
HOUR	SMALLINT	到着時刻 (時)
MINUTE	SMALLINT ARRAY[30]	バス到着時刻 (分)

6.2 データベース格納サンプルデータの生成条件

6.2.1 属性データの生成条件

ここでは、MAIN テーブルの ID、TYPE、NAME、ADDRESS、PHONE、ZIP、SINCE、REF_TABLE_NAME フィールド、BUS ××テーブルの HOUR、MINUTE フィールドに格納する属性データの生成条件とデータ制約について表 6.3 に示す。

表 6.3: 属性データの生成条件

列名	生成条件とデータ制約
ID	一意制約 (1,2,…,30000)
TYPE	【BUS、STATION、BANK、RESTAURANT、POLICE、HOTEL、HOSPITAL、CONVENIENCE、SCHOOL】からランダムに選択
NAME	最大長 30 のランダムな英文字列
ADDRESS	最大長 50 字以内のランダムな英数字列
PHONE	一意制約 (0852000001、…、0852090000)
ZIP	長さ 7 のランダムな数字列
SINCE	範囲 (1990-01-01〜1999-12-31) の日時型
REF_TABLE_NAME	最大長 12 のユニークな英文字列 (参照するバス停の時刻表テーブル名に対応)
HOUR	一意制約 (4,5,…,24)
MINUTE	範囲【0〜59】のランダムかつユニークな整数 ARRAY 型

6.2.2 空間データの生成条件１ (一様分布の場合)

TYPE-A-1 と同様なので 5.2.2 節参照の事。

6.2.3 空間データの生成条件２ (二次元正規分布の場合)

TYPE-B では、二次元正規分布の空間データを用意していない。

6.3 ワークロードの構成

ここでは、ワークロード TYPE-B のフローチャート、及びワークロードの定義をそれぞれ 6.3.1 節、6.3.2 節で述べる。尚、ワークロードの実装には、埋め込み C を利用している。

6.3.1 フローチャート

ワークロード TYPE-B のフローチャートを図 6.1 に示す。

(1)Generate Spatial Condition Routine の実行：(C 言語で記述)
「空間検索条件となる円の半径と中心座標の生成」
↓
(2)Spatial Search and Get Length Query の実行：(SQL で記述)
「MAIN テーブルに対し、(1) のルーチンで生成された空間検索条件で空間検索を実行し、
更に、目標物 ID 及び現在地から目標物までの距離を集計・抽出」
↓
(3)Get Min Length Location ID Routine の実行：(C 言語で記述)
「(2) のクエリーで抽出した目標物から現在地までの距離が最小である目標物 ID を抽出」
↓
(4)Get Array Table Query の実行：(SQL で記述)
「(3) のルーチンで取得した目標物 ID を利用して、Array 型の列が定義された BUS テーブルを抽出」

図 6.1: ワークロード TYPE-B のフローチャート

6.3.2 ワークロードの定義

●Generate Spatial Condition Routine：

図 6.1 の (1) に示すこのルーチンは、表 6.4 に示す生成条件に基づいて円の半径と中心座標 (X, Y) といった空間検索条件を生成している。このルーチンは、C 言語により実装している。

尚、このとき、(1) のルーチンで生成される円は必ず (1、1)〜(10000、10000) の範囲内に収まる (ちなみに、本ベンチマークの空間データは、(1,1)〜(10000,10000) の範囲内で生成している)。

表 6.4: 空間検索条件の生成条件 2

	生成条件
円の中心座標	一様分布 ((1001,1001)〜(9000,9000))
円の半径	正規分布 (平均 500、分散 $(30)^2$)

●Spatial Search and Get Length Query：

図 6.1 の (2) に示すこのクエリーは、(1) のルーチンにより生成された空間検索条件を利用して、空間検索を実行するクエリーである。このクエリーは、図 6.2 に示す SQL 文により実装した。

```
EXEC SQL
 SELECT ID, LENGTH2(current_x, current_y, LOCATION)
        into o_id, o_length
  FROM MAIN
    WHERE TYPE = 'BUS' AND
          WITHIN(LOCATION,condition) IS TRUE;
```

図 6.2: SQL 文による Spatial Search and Get Length Query(TYPE-B) の定義

ここで、図 6.2 において、(1) のルーチンに基づいて生成された空間検索条件は、埋め込み変数 condition を利用して C 言語から引き渡され、逆に SQL 文によるデータ抽出の結果は、それぞれ埋め込み変数 o_id、o_length に引き渡される。

また、図 6.2 の WITHIN 関数は、OGC が定める空間データ型に対して利用する関数である。また、図 6.2 の LENGTH2 関数は、第一引数、第二引数に現在地の X 座標、Y 座標を指定し、第三引数に空間データ型が定義されたフィールド名を指定する。このとき、LENGTH2 関数は入力された現在値と目標物との２点間の距離を抽出する関数で、本ベンチマークでは、これをストアドファンクションとして SQL 文により定義している (6.6 節参照の事)。

尚、TYPE-A の Spatial Search and Data Extraction Query(5.3.2 節参照) とは、検索条件に空間検索条件に加え、属性検索条件を指定している点が大きく異なる。

●Get Min Length Location ID Routine：

図 6.1 の (3) に示すこのルーチンは、(2) のクエリー内の LENGTH2 関数で抽出した目標物から現在位置までの距離が最小となる目標物 ID を抽出する。このルーチンは、C 言語により実装した。

●Get Array Table Query：

図 6.1 の (4) に示すこのクエリーは、(3) のルーチンで抽出した目標物 ID を利用して、MAIN テーブルの REF_TABLE_NAME フィールドに格納されている時刻表のテーブル名を取得し、SQL99 の新しいデータ型である Array 型が定義されたテーブルのデータを抽出する。このクエリーは、図 6.3 に示す SQL 文により実装した。

```
EXEC SQL
 SELECT MINUTE[1], MINUTE[2], ···, MINUTE[30]
        INTO  o_minute[1], o_minute[2], ···, o_minute[30]
  FROM (SELECT REF_TABLE_NAME FROM MAIN WHERE ID = min_length_id)
   WHERE HOUR = get_hour ;
```

図 6.3: SQL 文による Get Array Table Query(TYPE-B) の定義

図 6.3 において、抽出したいバスの到着時刻 (時) は、埋め込み変数 get_hour を利用して C 言語から引き渡され、逆に SQL 文による Array データの抽出は、埋め込み配列変数 o_minute に引き渡される。

6.4　ベンチマーク測定プログラムの構成

ここでは、本ベンチマーク TYPE-B の測定プログラムのフローチャート、ループ条件、及び測定結果については、それぞれ 6.4.1 節、6.4.2 節、6.4.3 節で述べる。

6.4.1　フローチャート

ベンチマーク TYPE-B の測定プログラムのフローチャートは、TYPE-A と同様なので図 5.8 を参照の事。尚、図 5.8 に示してある【ワークロード の定義】について、TYPE-B では、具体的に 6.3.1 節に示す流れでルーチンやクエリーを実行する。

6.4.2　ループ条件

TYPE-A と同様なので 5.4.2 節参照の事。

6.4.3　測定結果について

TYPE-A と同様なので 5.4.3 節参照の事。

6.5　テーブルの定義

MAIN テーブル、及び BUS ××テーブルは、それぞれ図 6.4、図 6.5 に示す SQL 文により定義した (図 6.4 の GEOMETRY 型が OGC の定める空間データ型である)。

また、ID フィールドは主キーとして定義し、空間データ型で定義した LOCATION フィールドには、空間インデックスを定義した (6.7 節参照)。

```
CREATE TABLE MAIN
(
ID               INTEGER       NOT NULL,
TYPE             VARCHAR(10)   NOT NULL,
NAME             VARCHAR(30)   NOT NULL,
ADDRESS          VARCHAR(50)   NOT NULL,
PHONE            VARCHAR(12)   NOT NULL,
ZIP              CHAR(7)       NOT NULL,
SINCE            DATE          NOT NULL,
REF_TABLE_NAME   VARCHAR(12)   NOT NULL,
LOCATION         GEOMETRY
)
PRIMARY KEY (ID);
```

図 6.4: SQL 文による MAIN テーブルの定義

```
CREATE TABLE BUS ××
(
HOUR    SMALLINT  NOT NULL,
MINUTE  SMALLINT  ARRAY[30]  NOT NULL
);
```

図 6.5: SQL 文による BUS ××テーブルの定義

6.6 ストアドファンクションの定義

図 6.2 で利用されている LENGTH2 関数の SQL 文によるストアドファンクションとしての定義を図 6.6 に示す。この関数の第一、第二引数には、現在位置の X 座標、Y 座標を指定し、第三引数には、空間データ型が定義されたフィールド名を指定する。この関数は、2 点間の距離を単純に二乗和した値を戻り値として返す。

尚、図 6.6 で利用されている X,、Y 関数は点の空間データに対して、それぞれ X 座標、Y 座標を取得する関数である。

```
CREATE FUNCTION LENGTH2(
  current_x   INTEGER,
  current_y   INTEGER,
  in_location geometry)
  RETURNS INTEGER
  BEGIN
    DECLARE xx, yy int;
    SET xx = X(in_location);
    SET yy = Y(in_location);
    RETURN (xx - current_x)*(xx - current_x)+(yy - current_y)*(yy - current_y)
  END;
```

図 6.6: SQL 文によるストアドファンクション LENGTH2 の定義

6.7 空間インデックスの定義

本ベンチマークでは、図 6.4 で定義した MAIN テーブルの LOCATION フィールドに対して、主キーである ID フィールドとは別のインデックスを図 6.7 に示す SQL 文により定義した。また、LOCATION フィールドに対するインデックス名は、TypeB_Spatial とした。

尚、空間インデックスに関する詳細は、5.6 節を参照の事。

```
CREATE INDEX TypeB_Spatial USING TYPE SPATIAL ON
MAIN(LOCATION) PLUGIN 'EXTENT=1,1,10000,10000;
DEPTH=5,PRECISION=1';
```

図 6.7: 空間インデックスの定義 2

6.8 実行環境

TYPE-A と同様なので 5.7 節参照の事。

第7章　本ベンチマークテストの実行結果

7.1　TYPE-A:エリアマーケティングの場合

本ベンチマークテスト TYPE-A(エリアマーケティング) の実行結果を表 7.1、表 7.2 に示す。

表 7.1: ベンチマークテスト TYPE-A のレスポンスタイム

Iteration	Response Time	
	TYPE-A-1	TYPE-A-2
1	5.92	5.29
2	5.69	5.20
3	5.61	4.45
4	5.73	4.22
5	5.53	5.13
6	5.77	4.53
7	5.63	4.86
8	5.69	4.66
9	5.83	4.31
10	5.69	4.73

表 7.2: Iteration1〜Iteration10 の平均レスポンスタイム、標準偏差、総レスポンスタイム

	TYPE-A-1	TYPE-A-2
Mean Response Time	5.71	4.74
Standard Deviation	0.11	0.36
Total Response Time	57.61	47.86

表 7.1、表 7.2 の実行結果は、ITERATION=10、 SEARCH_TIMES=100 の時の本ベンチマークテストの実行結果を示している。

7.2　TYPE-B:周辺位置情報システムの場合

本ベンチマークテスト TYPE-B(周辺位置情報システム) の実行結果を表 7.3、表 7.4 に示す。

表 7.3、表 7.4 の実行結果は、ITERATION=10、 SEARCH_TIMES=100 の時の本ベンチマークテストの実行結果を示している。

表 7.3: ベンチマークテスト TYPE-B のレスポンスタイム

Iteration	**Response Time**
	TYPE-B
1	12.29
2	10.69
3	10.41
4	9.83
5	8.36
6	8.05
7	8.36
8	8.23
9	7.98
10	8.14

表 7.4: Iteration1〜Iteration10 の平均レスポンスタイム、標準偏差、総レスポンスタイム

	TYPE-B
Mean Response Time	9.23
Standard Deviation	1.41
Total Response Time	92.86

第8章　検証

ここでは、7 章に示す本ベンチマークテスト TYPE-A、TYPE-B の実行結果を基にして、TYPE-A、TYPE-B の検証について述べる。

8.1　TYPE-A:エリアマーケティングについて

TYPE-A では、コンビニやファーストフードの出店計画を想定したエリアマーケティングにおいて頻繁に利用されると思われる空間検索の処理性能を主な測定対象とした。その実行結果においては、TYPE-A-1 と TYPE-A-2 において多少の違いが見られた。

まず、平均レスポンスタイムの違いに対する原因については、一回のデータベース・トランザクション (空間検索) 処理により抽出される TYPE-A-1 の顧客レコード数が、TYPE-A-2 の顧客レコード数よりも多いためだと思われる。即ち、一回のワークロードの処理量も TYPE-A-1 の方が TYPE-A-2 よりも多くなるため、TYPE-A-1 の平均レスポンスタイムが TYPE-A-2 よりも多少長くなっている。

一方、標準偏差の違いに対する原因については、TYPE-A-1 の空間データが一様分布であるのに対し、TYPE-A-2 の空間データが疎と密の部分が混在する二次元正規分布であるためだと思われる。このため、TYPE-A-2 の方が TYPE-A-1 よりも標準偏差が大きい、つまり空間検索処理の実行時間のちらばりが大きいという事が表 7.1, 表 7.2 の結果から読み取れた。

8.2　TYPE-B:周辺位置情報システムについて

TYPE-B では、モバイル端末と GPS 衛星を利用して取得した現在位置から近隣の位置、及び関連情報の取得処理を想定した周辺位置情報システムにおいて頻繁に利用されると思われる空間検索の処理性能を主な測定対象とした。その実行結果については、TYPE-A と TYPE-B において、平均レスポンスタイムに大きな違いが見られた。

まず、平均レスポンスタイムの違いに対する原因については、TYPE-B(１テーブル× 3 万レコード＋１００テーブル×２１レコード) の方が、TYPE-A(１テーブル×９万レコード) よりもデータベース・サイズが大きいためだと思われる。また、TYPE-B の方は、検索条件に空間検索条件と属性検索条件の２つを指定し、その後 ARRAY 型を定義したテーブルのデータを抽出しているため、ワークロードの処理にかかる時間が TYPE-A に比べ多くなっていると考えられる。

8.3　総括

TYPE-A と TYPE-B の考察結果をまとめると、空間検索を対象としたベンチマークテストでは、空間データの生成モデル、テーブルの構成、及び空間検索を伴うデータベース・トランザクションのタイプの選択などといったワークロードの構成が、ベンチマークの測定結果に大きな影響をおよぼすことが本ベンチマークの測定結果から伺えた。

第9章　結論

本研究では、今回エリアマーケティング、及び周辺位置情報システムという２つタイプの空間検索を対象としたORDBMSベンチマークテストを開発し、その仕様と実装について述べてきた。

結果的に、本研究では、今までにない空間検索を対象としたORDBMSを開発できたということで、0から１への作業を行えたと思う。今後は、次に述べる幾つかの課題を考慮していくことで、本ベンチマークを１から２、そして２から３へと更に拡張させていけると思う。

その今後の課題としては、空間検索をターゲットとするワークロードのバリエーションを増やしていくことが挙げられる。また、周辺位置情報システムを利用した空間検索の利用では、最短経路問題に対する処理性能をどう評価するかも今後の課題に挙げられると思われる。そして、マルチユーザー環境に対応し、かつHTTPサーバーとの性能を総合的に評価できるベンチマークに拡張していく事も、今後の課題として挙げられるだろう。

今回のこの研究が、今後のORDBMSベンチマーク開発に一石を投じ, 更なるORDBMSベンチマーク開発の推進と発展に繋がっていけば幸いである。

謝辞

本研究にあたり、最後まで熱心な御指導をいただきました田中章司郎先生には、心より御礼申し上げます。

また、田中研究室生の森本誠人君、江野本隆行君、森下旭君、荒木俊太郎君、富岡俊幸君、中村吉郎君、平田純一君、辰己圭介君、貫目洋一君、驛範之君、高木明君、森岡慶彦君には、本研究に関して数々の御協力と御助言をいただきました。厚く御礼申し上げます。

なお、本論文と本研究で作成したプログラム等のすべての著作権を田中章司郎教授に譲渡致します。

引用文献

[1]G.Bai、H.Amano、A.Makinouchi、
WAKASHI/C:Strage System for Multimedia Database、
Systems and Computers in Japan、Vol.26、No.7、pp.13-22、July、1995

[2]R.G.G.Cattell、J.Skeen、
Object Operations Benchmark、
ACM Transactions on Database Systems、Vol.17、No.1、pp.1-31、March、1992

[3]Jim Gray、喜連川 優、渡辺 榮一 (監訳)、
データベース・ベンチマーキング、
日経 BP 社、pp.328、December、1992

[4]OpenGIS Consortium、
Simple Feature Specification for SQL Revision1.1、
OpenGIS Project Document 99-049、May、 1999

[5] 田中 章司郎、
アプリケーション・パッケージ SQL/MM 標準化動向、
電子情報通信学会技術研究報告、Vol.99、No117(DE99-4)、June、1999

[6]ISO/IEC13249-3、
Infomation technology Database language-SQL Multimedhia and Application Packages -Part3:Spatial、
pp.1-332、1999

[7] 日立ソフトウェアエンジニアリング、
HiRDB 空間検索プラグイン、
pp.1-199、2001

[8]Chris Loosley、Frank Douglas、間宮 あきら (編)、
データベースチューニング (上)、
日経 BP 社、pp.306、June、1999