関数的検索機能を有する地理情報システム　MILES

松山　隆司，三根　清，レ・ヴェト・ハオ，長尾　真
（京都大学　工学部）

1．はじめに

地図データは，社会的，行政的あるいは科学的な調査や分析，計画などの目的のために広く用いられている。近年，地図情報をデータベース化し，情報の管理・検索を効率よく行いたいという要求が高まって来ており，さまざまな方式の地理情報システムが提案されている。[1]-[5]

従来，各種の地理的情報を計算機で処理するためのデータ構造として，メッシュ法が広く用いられてきた。[5] しかし，この方法では，(1)線的な対象（道路，川，鉄道）に関する情報をうまく表現できない。(2)解析の対象地域や調査区域とメッシュとの整合性がとりにくい。といった欠点があり，処理結果の信頼性，解釈の容易性に問題が生じる。このため，最近では，線図形として描かれた地図の持つ情報をそのままの形で計算機に蓄え，検索するという観点から，点・線・面の接続関係に基づいたいわゆるトポロジカル・ネットワークによって地図を表現する方法がよく用いられる。[1]-[4]

地理的情報を大別すると，(1)図形的情報，(2)非図形的情報　に分けられる。さらに，(1)の情報は，(1-a)位相幾何学的情報（隣接・交差・包含）と(1-b)距離的情報（近接・平行）に分けられる。このうち，(2)と(1-a)の情報の構造化は，一般のデータベースシステムの枠組でも取り扱うことができ，実際に関係データベースを利用した地理情報システムがいくつか開発されている[4],[5] しかし，一般のデータベースシステムの上に1つのアプリケーションとして地理情報システムを作成するという方法では，処理効率や上の(1-b)の情報の構造化，あるいは地理情報システムで要求される各種の図形操作・表示機能といった点で問題が生じる。このような観点から，我々は，トポロジカル・ネットワークに基づき，各種の図形操作が柔軟に行える地理情報検索専用システムMILESを開発することにした。

我々が地理情報システムMILESの開発に際して特に問題とした点は，

(i) 同一地域に対する複数の目的別主題図を統一的に表現し，異なった主題図に関係する検索要求を柔軟に処理するための方式の開発

(ii) 各種の図形操作や表示，情報検索に関する要求を簡潔でわかりやすい形式で表現できる情報検索用言語の開発

(iii) 対象間の近接性（先の(1-b)の情報）に基づく検索要求を効率良く処理するための方式の開発

である。

以下では，まず2においてシステムの構成を示し，3～5において上に挙げた問題に対して我々が採用した方式について述べる。

2．システムの構成

図1にMILESの構成を示す。MILESにおける検索は，京都大学大型計算機センター（M200）のTSS環境下で，我々が開発した関数的検索用言語GPLを用いて行われる。検索結果はGPLのグラフィック関数を用いることによって，カラーグラフィック装置（Tektronix 4027），あるいは通常のベクトル型のグラフィック装置（Tektronix 4015, 4010）上に図形として表示することができる。

ファイルシステムは可変長レコードが取り扱えるように新たに開発したもので，図形データファイルに対しては，図形要素（点・線・面）の空間的近接性が2次記憶装置上でも保たれるような特殊な構造化が行われ（4参照），効率的なファイルアクセスが実現できるように工夫されている。

データベース管理システムは，いわゆるネットワーク型のモデルに基づいて設計されており，レコード型の定義，特定のレコードの読み出し・書き込み，レコード間の関係を表わ

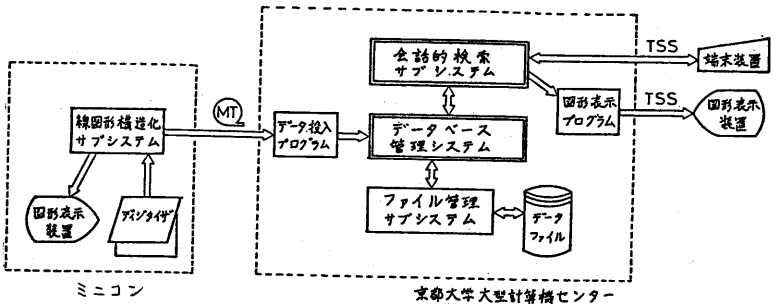


図1　システムの構成

すリンクの定義，リンクを利用したレコードの検索，など一般的なデータベースの情報検索，管理機能を持っている。

図形データの構造化（地図のディジタル化および点・線・面による構造記述の作成）はディジタイザやグラフィック表示装置を用いたミニコン Interdata 8/32 上の会話型図形編集システムを利用して別途行われる。構造化された図形データと各種の（非図形的な）地理的情報が，データ投入プログラムによってMILESに蓄えられる。

3. 地図モデル

1. で述べた(i)の問題に対して，MILESでは図2に示すような地図モデルを用いた。このモデルでは，異なった複数枚の主題図を重ね合わせて得られる地図（基本図，図3参照。）を地図モデルの基本構造とし，各主題図は基本図から誘導されたものとして表現される。

3.1. 基本図構造

各主題図における地理的対象（地理実体）を表わす図形（線・面）は，主題図の重ね合せにより，他の地理実体を表わす図形によって小さな線分や面分に細分される（図3参照）。こうして生成される線分・面分を基本図線・基本図面と呼ぶ。* また，点型の地理実体を表わす点や線の端点，あるいは線の交差によって生じた交点を基本図点と呼ぶ。各基本図点・線にはその位置や形状を表わす座標値が属性として与えられ，さらに，基本図線にはその長さ，基本図面にはその面積が属性として書かれる。

基本図構造では，基本図点・線・面といった図形要素間の位相幾何学的な関係が，以下に示す6つの接続関係を用いて統一的に記述される。（図4参照。）

(a) PTOL：基本図線 ⟶ 基本図点
各基本図線には固有のお向が与えられており，基本図線の始点と終点となる基本図点を示す関係

(b) LTOP：基本図点 ⟶ 基本図線
基本図点を終点，始点とする基本図線を示す関係で，反時計回りに基本図線を並べる。この際，その基本図点を終点とする基本図線には"－"の符号を付ける。

(c) RTOL：基本図線 ⟶ 基本図面
基本図線の左右にある基本図面を示す関係。

(d) LTOR：基本図面 ⟶ 基本図線
基本図面の境界線を構成する基本図線を示す関係。外境界線は反時計回り，内境界線は時計回りに，それぞれ境界線に含まれる基本図線を並べあげる。この際，境界線の向

* 基本図線は分岐・交差のない線分（直線とは限らない），基本図面は他の基本図面と共有部分を持たない最小の単連結の面

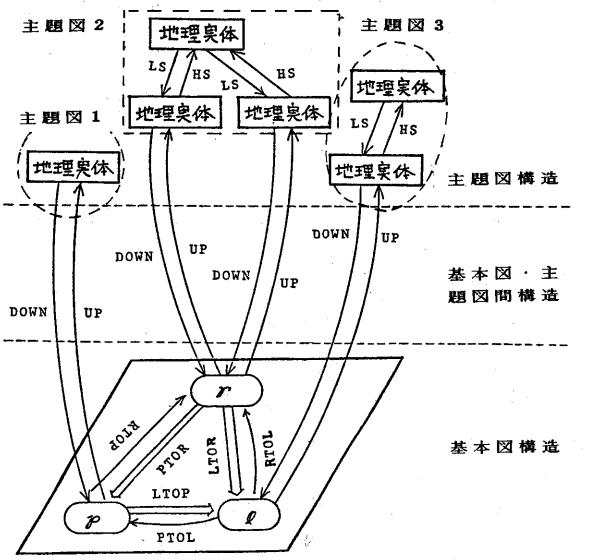


図2　地図モデル

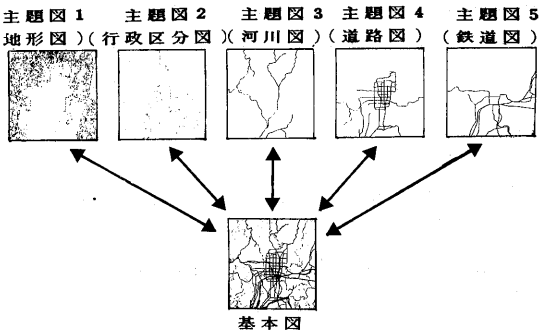


図3　主題図と基本図

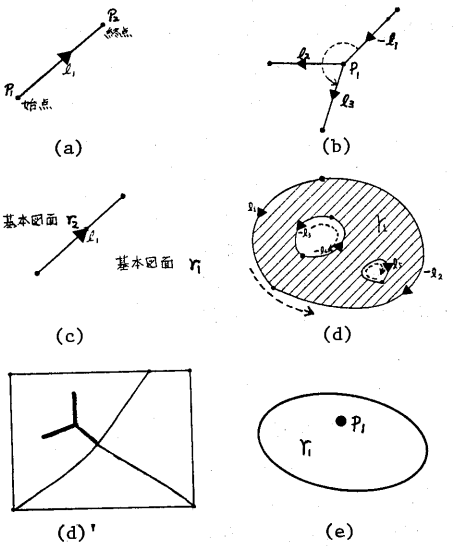


図4　図形要素間の接続関係

きと反対の方向を持つ基本図線には"－"の符号を付ける。また，基本図面に入り込む線は縮退した内境界線として考える。(図4(d)')

(e) PTOR：基本図面 ⟶ 基本図点
基本図面に孤立して含まれる(基本図線の始点終点ではない)基本図点を示す関係。

(f) RTOR：基本図点 ⟶ 基本図面
孤立した基本図点からそれを含む基本図面を示す関係。

3.2 基本図・主題図間構造

このレベルでは，関係UPとDOWNによって主題図中の地理実体と基本図中の図形要素とが結ばれる。地理実体は関係DOWNの行き先によって点型，線型，面型に分かれる。いずれの場合も，地理実体の図形的構造は基本図点，線，面を要素とする集合(線型の場合は順序集合)として表わされる(図5)。

図5に示したように，一般に基本図中の1つの図形要素は複数の地理実体に対応している。すなわち，このモデルでは，1つの図形要素に複数の意味付けを与えることができ，「道路の下を地下鉄が走っている。」などといった複合情報を整合性のとれた形で蓄え表現することができる。例えば，図5(b)において，基本図線 l_2 が道路 L_1 の一部でもあるし，地下鉄 L_2 の一部でもあるとすればよい。また，面型の地理実体間の包含関係や共通部分の切り出しなどは，基本図面を要素とする単純な集合演算で容易に求めることができる。

3.3 主題図構造

このレベルでは，地理実体の持つ属性や各種の非図形的な関係がネットワーク型のデータ構造で表現されており，現実世界のさまざまな地理的情報が記述される。地理実体のクラス間の関係のうち，行政区に見られるような図形としての包含関係と1対1に対応するものは，HS，LSという特別なリンクによって作られる階層構造として表現される。すなわち，"市"というクラスに属す地理実体"京都市"の図形的構造は，直接基本図面の集合として表現されるのではなく，関係LSによって一担"区"というクラスの地理実体の集合に展開され，その後関係DOWNによって基本図面の集合に展開されることになる(図2参照)。

4. 空間的近接性に基づく図形ファイルの構造化

基本図構造で記述される図形情報は位相幾何学的なもののみで，図形間の距離に基づく関係は表現されていない。図形の持つ距離的情報のうち，対象間の近接性に関する情報は，「地点Aに最も近い～」や「地点Aから半径r以内の～」といった形の検索要求において，中心的な役割を果し，かつその要求頻度も高い。ところが，これまでに開発された地理情報システムのほとんどのものでは，この情報がうまく表現されておらず，上のような検索要求に対して効率良く処理を行うことができない(先の(iii)の問題)。

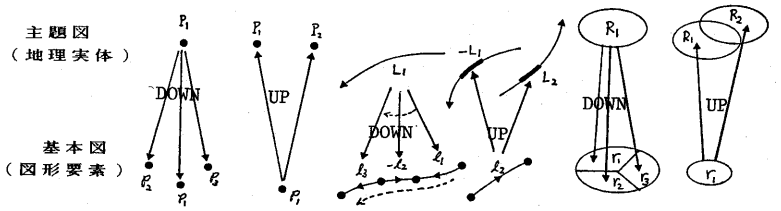
MILESにおいては，対象間の近接性を直接データ構造として表現するのではなく，地図空間における図形要素の近接性をファイル構造に反映させることにより，近接性に関連した各種の検索要求を効率的に処理できるように工夫した。すなわち，図6に示すように，(基本図)地図空間を座標軸に平行な線分によりいくつかの矩形領域に分割し，各矩形領域に含まれる図形要素のレコードを2次記憶上の同一のページに格納するというファイルシステムを用いた。この方法によると，空間的に近い図形要素のレコードが2次記憶上でページ単位にクラスタ化されることになり，1つのページをアクセスすることが，地図空間中の矩形領域をアクセスすることと同じ意味を持つ。この結果，ある対象からある距離だけ離れた対象を構成する図形要素のレコードがどのページに含まれるかが容易に計算でき，無駄な2次記憶へのアクセスをなくし，処理を高速化できる。

こうしたファイルの構造化は，次のような手順で，基本図構造を記憶するファイルに対して適用される。

(1) 各基本図点・線・面に対してその図形要素の代表点(外接長方形の重心)を求める。

(2) 地図空間上に分布する代表点の集合を Fe-d tree 法[7]により分割する(図7)。

(i) 分割線の両側の点の数が等しくなるような位置で，矩形領域(最初は地図空間全体)の長辺を



(a) 点型　(b) 線型　(c) 面型

P_1: $(p_1\ p_2\ p_3)$　L_1: $(l_1\ -l_2\ l_3)$　R_1: $(r_1\ r_2\ r_3)$

図5　関係UPとDOWN

2分割する。

(ii) 分割されてできた2つの矩形領域にそれぞれ含まれる点の数(図形要素のレコードの総量)が，ページの容量以下になるまで，(i)の操作を各矩形領域に対して再帰的に適用する。

(3) 最終的に得られた矩形領域に含まれる代表点に対応する図形要素のレコードの集合を2次記憶の1つのページに格納する。

地図空間の分割状態は，各矩形領域を葉ノードとする2進木で表現され，この木構造を用いてページのアクセスを制御する。例えば，「図6において，太い矩形内の点をすべて求めよ」といった範囲探索に答えるには次のようにすればよい。まず，木の根ノード(地図空間全体を表わす)に書かれている分割線の位置から，問題となる範囲が左右いずれの部分木(空間的には分割線Aの左右いずれかの矩形領域)と重なりがあるかを調べる。今の場合，問題の範囲は右の部分木のみと重なっており，右の枝をたどる。(左の部分木はたどらなくてよい。)同様の操作を各部分木に対して逐次反復し，最終的にどのページ(葉ノード)が問題の範囲と重なりがあるかを調べる。そののち，必要なページ(図中の"S"印)を2次記憶から読み込み，その中のデータのそれぞれが問題の範囲内にあるかどうかを調べる。この例では，結局3回の2次記憶へのアクセスと6つの点のレコードの比較で要求が満たされる。これは，何ら構造化を行わず全数チェックを行う場合(16回の2次記憶アクセスと32点のレコードの比較)に比べ大幅な高速化となる。(各種分割法の性能評価および線型，面型の図形要素の取り扱いに関しては[6]を参照。)

算述演算関数
+，−，*，%(DIVIDE)，|(ABSOLUTE)， @CONVERT(CONVERT DATA TYPE OF NUMERAL)
論理演算関数
&(AND)，|(OR)，¯(NOT)，=，¯=(NOT =)， <，>，<=，>=
集合演算関数
@SET(ELIMINATE DUPLICATION)，@UNION， @AND，@DIFF(DIFFERENCE)， @NEUT(ELIMINATE +/− ELEMENT PAIR)
リスト処理関数
@IOTA(GENERATE LIST)， @RHO(COUNT THE NUMBER OF ELEMENTS IN A LIST)， `，`(CONVERT COMPLEX LIST TO SIMPLE LIST)， `「n」`(SELECT n-th ELEMENT)， ；(XCONS)，ETC.
図形操作関数
@LTOP，@PTOL，@RTOL，@LTOR， @RTOP，@PTOR　(本文参照)
データベース操作関数
@READREC(READ RECORD)，@WRITREC (WRITE RECORD)，@DELREC(DELETE RECORD)， @MAPPING(本文参照)，ETC.
グラフィック関数
@MOVE(MOVE BEAM)，@DRAW(DRAW LINE)， @TEXT(PRINT TEXT)，@GRA(SPECIFY GRAPHIC OPTIONS, PAINT AREA ETC.)

表1　GPLの組み込み関数

5. 関数的検索用言語GPL

ここでは，MILESにおける検索用言語として開発したGPLの各種機能および検索例について述べる。GPLは，APLを基にした解釈実行型の関数型言語で，複雑な地図情報の検索手順をいくつかの関数の組み合せとして表現できる。このような関数型言語を利用すると，(i)検索の手順を明瞭な形式で記述することができ，検索過程が概念的に理解しやすくなる。(ii)複雑な処理手順を1つのグローバルな関数として定義することにより，エンドユーザーにとって使いやすいコマンド群を作成することができる。といった利点がある。さらに，GPLでは入力された式が直ちに解釈実行され，結果が出力されるので，会話的に検索を進めることができる。

GPLでは，基本的なデータタイプとして1次元配列(線形リストや集合を表わすものと見なすこともできる)を用いている。GPLの関数には，システムの組み込み関数とユーザーが複

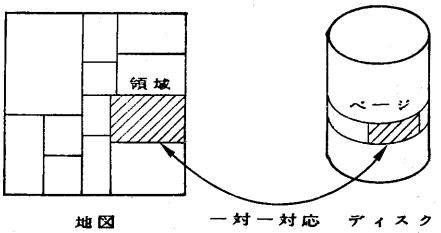


図6　領域とページの対応

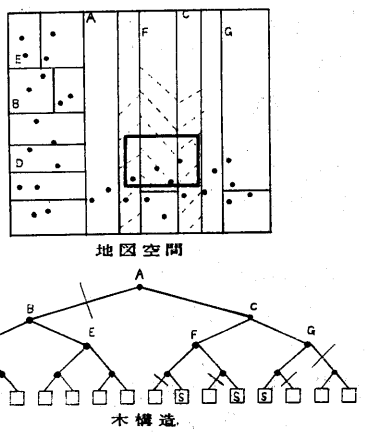


図7　分割木を用いた範囲探索[7]

数の式をまとめて1つの関数として定義する定義関数の2種類がある。表1に組み込み関数のタイプと代表的な関数の例を示す。GPLは，ASCII文字セットを用いており，特殊記号や"@"付きの名前によって組み込み関数が表現される。また，式の評価は式の右側から左に向けて行われ，評価の順序を特に指定する場合には，通常の算術式と同様に，"("と")"を用いる。さらに，GPLにはいくつかの制御文や入出力文があり，これらを用いて手続きを作成することも可能である。

以下では，3で述べた地図モデルの各階層における図形操作・検索の方式について述べる。

5.1 基本図構造における図形操作

基本図構造では，3.1で述べた6つの接続関係によって基本図点・線・面の位相幾何学的構造が記述されている。GPLには，これら6つの接続関係にそれぞれ対応する図形操作用関数が組み込まれている。以下では，図8を例としてこれらの関数の機能を説明する。

(a) @PTOL：基本図線を入力とし，その終点である基本図点を出力する。(始点を出力しない理由は後述する。) 基本図線の始点を求めたい場合は，基本図線に"−"の符号を付けて入力する。

例：$p_4 \xleftarrow{@PTOL} l_3$
$p_6 \xleftarrow{@PTOL} -l_3$

(b) @LTOP：基本図点を入力とし，その点を始点，終点とする基本図線のリストを出力する。基本図線の並べかは反時計回りの順序で，その基本図点を終点とする基本図線には，"−"を付ける。

例：$(-l_3\ -l_6\ l_1\ l_5) \xleftarrow{@LTOP} p_4$

(c) @RTOL：基本図線を入力とし，その線の向きに対して右側の基本図面を出力する。@PTOLと同様，左側の面を求めるには，基本図線に"−"を付ける。

例：$r_3 \xleftarrow{@RTOL} l_3$
$r_1 \xleftarrow{@RTOL} -l_3$

(d) @LTOR：基本図面を入力とし，その境界線を構成する基本図線を出力する。出力の形式は
((外境界線)(内境界線1)…(内境界線n))
で，外境界線は反時計回り，内境界線は時計回りに基本図線を並べる。

例：$((l_1\ -l_2\ l_3)(l_4)(l_5\ -l_5)) \xleftarrow{@LTOR} r_1$

(e) @PTOR：基本図面を入力とし，その面に孤立して含まれる基本図点を出力する。

例：$p_0 \xleftarrow{@PTOR} r_3$

(f) @RTOP：孤立した基本図点を入力とし，その点が含まれる基本図面を出力する。

例：$r_3 \xleftarrow{@RTOP} p_0$

関数@PTOLにおいて，出力を基本図線の終点のみとした理由は，図9のような線を基本図点 P_1 から追跡する場合，関数として@LTOP，@PTOLを交互に適用してゆくだけでよく，図形操作が簡潔に記述できるためである。つまり，始点，終点が同時に出力されるとたえずいずれを選ぶかをプログラムで書かねばならず，それだけ手順が複雑になる。また，@RTOLにおいても同様の理由によって，基本図線の右側の基本図面のみを出力するようにしてある(次の例のステップ(iii)参照。)。

一般に基本図構造における図形操作には，1対nの対応付け，符号属性，順序構造が含まれ，上記の6つの関数と各種の集合演算やリスト処理用関数を組み合わせることによりさまざまな図形操作が実現できる。

例えば，「図8の基本図面 r_1 に外側で隣接する基本図面を求めよ。」という要求に対しては，以下のような関数を逐次適用してゆけばよい。

(i) r_1 の境界線を求める。
$((l_1\ -l_2\ l_3)(l_4)(l_5\ -l_5)) \xleftarrow{@LTOR} r_1$

(ii) リストの第1要素(外境界線)を選ぶ
$(l_1\ -l_2\ l_3) \xleftarrow{SELECT\,1} ((l_1\ -l_2\ l_3)(l_4)(l_5\ -l_5))$

(iii) 境界線の反対側の基本図面を求める。
$(r_2\ r_3\ r_3) \xleftarrow{@RTOL\#} (l_1\ -l_2\ l_3)$

(iv) 要素の重複を除く
$(r_2\ r_3) \xleftarrow{@SET} (r_2\ r_3\ r_3)$

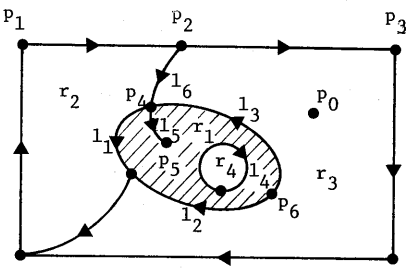


図8 基本図の例

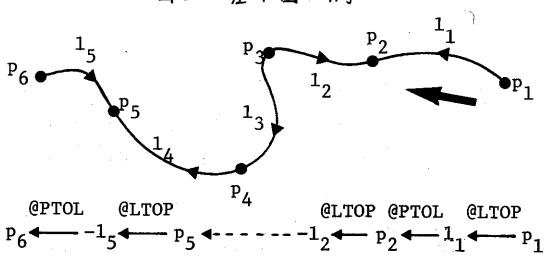


図9 関数@LTOP，@PTOLによる線の追跡

ここで，"#"は関数をリストの要素毎に適用するための拡張機能を示す。(i)～(iv)の処理を1つにまとめ，新たな定義関数O_AJAC_Rを作るには次のようにすればよい。

```
$ R_SET @= O_AJAC_R  P_AREA
[1]R_SET@=@SET@RTOL#SELECT1@LTOR P_AREA
$
```

ここで，"$"は関数定義の開始終了記号，1行目は関数名(O_AJAC_R)，入力変数(P_AREA)，出力変数(R_SET)の宣言で，[1]の行が関数の本体である。また，"@="は代入記号である。このように，一担関数O_AJAC_Rを定義すると，以後はこれを組み込み関数と同様にして使うことができ，さらに複雑な図形操作を簡潔に記述することが可能となる。

5.2 基本図・主題図間構造による地理実体と図形要素の対応付け

図5に示したように，地理実体の図形的構造は，基本図における図形要素の集合として表現され，両者の間は関係UPとDOWNによって対応付けがされている。

関数DOWN(定義関数である)は，地理実体を入力とし，出力としてそれを構成する図形要素のリストを返す。例えば，図5(b)の地理実体L_1に対して関数DOWNを適用すると，$(l_1\ -l_2\ l_3)$が返される。従って，地理実体間の図形的な関係を求めるには，

(i) 各地理実体を関数DOWNによって図形要素の集合に展開する。

(ii) 図形要素の集合に対し，集合演算やリスト処理，あるいは5.1で述べた図形操作用関数を適用する。

といった手順を取ればよい。

例1. 面型の地理実体R_1とR_2の共通部分を求める。(図10(a))

(i) R_1, R_2をそれぞれ基本図面の集合に展開する。

$$(r_1\ r_2) \xleftarrow{DOWN} R_1\ ,\ (r_2\ r_3) \xleftarrow{DOWN} R_2$$

(ii) 集合間の共通要素を求める。

$$(r_2) \xleftarrow{@AND} (r_1\ r_2),\ (r_2\ r_3)$$

ここで@ANDは2引数の関数で

リスト1 @AND リスト2

の形式で用いられ，2つのリストの共通要素を結果として返す。

例2. 面型の地理実体Rに含まれる線型の地理実体Lの部分を求める。(図10(b))

(i) R, Lを図形要素に展開する。

$$(r_1\ r_2) \xleftarrow{DOWN} R,\ (l_1\ l_2\ l_3) \xleftarrow{ABSOLUTE} (l_1\ -l_2\ l_3) \xleftarrow{DOWN} L$$

(ii) R内に含まれる基本図線(境界線も含む)を求める。

$$(((l_4\ l_2))\ ((-l_2\ -l_5))) \xleftarrow{@LTOR\#} (r_1\ r_2)$$

$$(l_4\ l_2\ -l_2\ -l_5) \xleftarrow{DIRECT\_SUM} (((l_4\ l_2))\ ((-l_2\ -l_5)))$$

$$(l_4\ l_2\ l_2\ l_5) \xleftarrow{ABSOLUTE} (l_4\ l_2\ -l_2\ -l_5)$$

$$(l_4\ l_2\ l_5) \xleftarrow{@SET} (l_4\ l_2\ l_2\ l_5)$$

(iii) (i)と(ii)で得られた基本図線の集合の共通要素を求める。

$$(l_2) \xleftarrow{@AND} (l_1\ l_2\ l_3),\ (l_4\ l_2\ l_5)$$

ここで関数ABSOLUTEとDIRECT_SUMはそれぞれシステムの組み込み関数で，表記上は"("と","によって表現される。

DOWNの逆関数HS(定義関数である)は，基本図構造中の図形要素を入力とし，それが属す主題図構造中の地理実体を出力する。図5に示したように，一般に1つの図形要素は複数の地理実体に対応している。このため，現在の関数HSでは，ある図形要素が異なったクラスの地理実体に対応している場合には，それらのクラス名を表示し，ユーザーにその内の1つを選択させる。また，図形要素が同一のクラスの複数の地理実体に対応する場合は，それらをリストとして返す(図12参照)。

5.3 主題図構造におけるデータベース操作

主題図構造では，地理実体の持つ非図形的情報がネットワーク型のモデルで表現されており，GPLには，このモデルに基づいた情報検索用関数がいくつか用意されている。

まず，指定された属性を持つ地理実体のレコードを検索するための関数としてSELECTがある。これは，SELECT 属性名リスト FROM 地理実体(クラス)名 WHERE 条件式の形式をとり，指定されたクラスの地理実体で条件を満たすものを選び，それらの地理実体の指定された属性の値を返す。例えば，

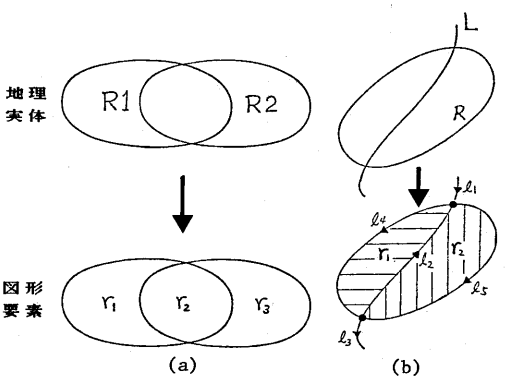


図10 地理実体間の図形的関係

SELECT "*" FROM 'KU' WHERE 'FS "NISHIGYOKU" = FQ "NAMAE"'

とすると，"NAMAE"という属性が"NISHIGYOKU"である地理実体を"KU"というクラスから選び，その識別番号を返す．

一般に，地理実体のクラス間には，それらの間の非図形的な関係を示す名前付きのリンクが張られている。このリンクに基づく検索を行うための関数として@MAPPINGがある。これは，リンクの名前と地理実体の集合とを入力とし，集合中の各地理実体とそのリンクで結ばれている地理実体をリストとして返す(図11)。

5.4 検索例

5.1～5.3で述べた各種の関数を組み合わせることにより，さまざまな地理的情報を検索することができる。GPLには，検索されたデータを図形として表示するための関数がいくつか用意されており，検索結果にこうしたグラフィック関数を適用することによって，カラーグラフィック装置に結果を図形として表示することができる。例えば，関数DRAWは，地理実体の識別番号を入力とし，その図形的表現を表示装置に出力する。

京都市の行政区分図，道路図，河川図の3枚の主題図からなるデータに対する検索の例を図12に示す。ここで，検索要求は次のものである。

(a)「西京区」を表示せよ。
(b) 西京区に接する川を求め表示せよ。
(c) 西京区を通過する道路を求め表示せよ。
(d)「桂川」を求め，それが通過する区を求め表示せよ。

6. おわりに

本報告では，我々が開発した地理情報システムMILESの各種機能について述べた。本システムの開発の目的は，1で述べた問題(i)～(iii)に対する効率的な処理方式を開発することにあり，これらの点に関しては，一応の成果が得られた。しかし，MILES

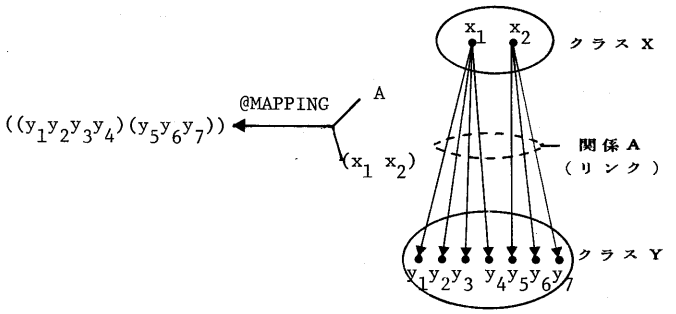
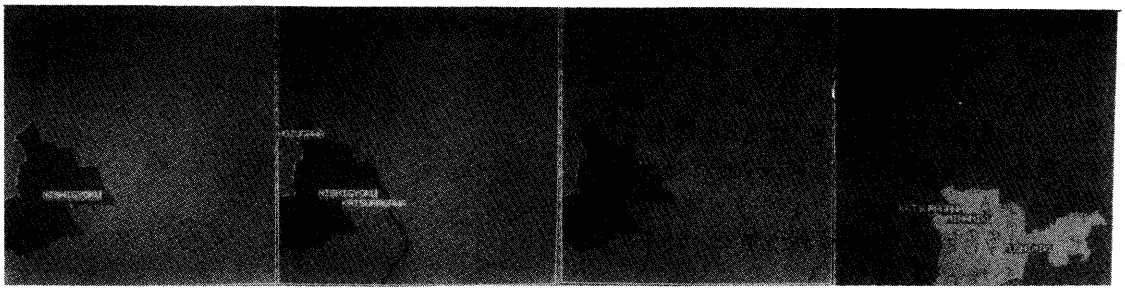


図11 関数@MAPPING



(a) (b) (c) (d)

```
:PIC1 @= SELECT "*" FROM 'KU' WHERE 'FS ''NISHIGYOKU''=FQ''NAMAE'''
:DRAW PIC1 ----------(a)
:PIC2 @= HS ADJAC_L PIC1
( KASEN_KUBUN   DOURO_KUBUN )------
( NANBANME O ERABI MASUKA?)
[]<-:1
( KASEN_KUBUN        5       4)
( IS IT OK ?)
[]<-:'NO'
( KAWA               2       10)
( IS IT OK ?)
[]<-:'YES'                   ------
:DRAW PIC2 ----------(b)
:DRAW HS TRAV_L PIC1 ------(c)
:DRAW HS TRAVERCE SELECT "*" FROM 'KAWA' 'FS''KATSURAGAWA''=FQ''NAMAE'''---(d)
```

関数HSからの質問
(以下の例では省略してある)

注) 関数 ADJAC_L, TRAV_L, TRAVERCE は，それぞれ線型と面型の地理実体間の図形的関係を求めるための定義関数である。

図12 検索例

を実用的なシステムとするためには，次のような点で拡張や改良を行う必要がある。

(1) 地図モデルの拡張：現在のモデルでは，1つの主題図に含まれる地理実体はすべて同じ型（点，線，面）に限られており，交差点と道路といった異なった型の地理実体を同一の主題図として見ることができない。主題図としての地理実体のまとめ方やその効果に関しては，今後研究の余地が大いにある。さらに，道路の右左折禁止などといった履歴性（どの道路から交差点に入ったか）を有する情報の表現法に関しても検討する必要がある。

(2) GPLの拡張：現在のGPLには，ユーザーの定義した関数をその入力引数のリストの要素毎に適用するという拡張機能がなく，検索手順をより簡潔に記述するためには，この点に関しての機能の拡張が必要である。また，GPLは解釈実行型の言語であるため，処理時間がかなりかかり，GPLの高速化も1つの課題である。

(3) データの入力・修正：現在データの入力は人手で行っており，作業量が多い。これを自動化することは実用化に大きく進むことになる。また，現在のシステムは検索を主としたものとなっており，図形データの追加・修正をGPLを用いて行うことができない。このためには，会話的な図形編集機能をGPLに追加することが必要である。

(4) 対象データの拡張：本システムに入力される主題図は，すべて同縮尺のものに限られており，縮尺の異なった地図の取り扱いや，新たに主題図を追加する場合，主題図間の整合性をいかに保つかという問題がある。さらに，メッシュデータと地図とを融合させ，より多角的に地理情報を処理できるようにすることも今後の課題である。

［参考文献］

[1] First International Advanced Study Symposium on Topological Data Structures for Geographic Information Systems, Harvard Papers on Geographic Information Systems, 1978.

[2] Nagy, G. and Wagle, S., Geographic Data Processing, ACM Computing Survey, Vol. 11, No. 2, 1979, pp. 139－182.

[3] 中森，地理情報を計算機で扱う技術の現状と問題点，オペレーション・リサーチ，1978年4月，pp. 240－248.

[4] 松家，杉本，宇土，地域計画のための地理的・社会的データベース，情処学会研資，データベース管理システム 22－2，1980.

[5] 根本，長沢，小出，国土数値情報利用・管理システム（ISLAND）の開発，FUJITSU, Vol. 32, No. 2, 1981, pp. 178－185.

[6] 松山，レー・ヴェト・ハオ，吉田，長尾，空間的近接性に基づくファイル分割アルゴリズムの性能評価，信学技報，IE 81－14, 1981.

[7] Bentley, J. L. and Friedman, J. H., Data Structure for Range Searching, ACM Computing Surveys, Vol. 11, No. 4, 1979, pp. 397－409.