ルックアップ・テーブルにより探索領域を限定した日本全国道路網における経路探索手法

飯村 伊智郎† 加藤 誠巳†

近年車載ナビゲーション・システムの普及が著しいが，目的地までの最適経路を提供するものは少ない．最適経路を算出するためには道路ネットワーク・データを必要とするが，例えば(財)日本デジタル道路地図協会の作成した日本全国の基本道路は約23万個のノード，約52万本のリンクより成り，その経路探索用データのデータ容量は約8MBと大きく，ディスクへのアクセス・読み込み時間，探索を実行するために必要なRAM領域，ならびにCPU時間が増大し，実用に耐えなくなることがある．従来からこれらの問題点を解決するためいくつかの方法が提案されているが，種々の問題点があった．本論文では日本全国の道路ネットワークを複数個の領域に分割し，任意の2つの領域間の最適経路を算出するために必要にして十分な領域の集合をオフラインで算出してテーブル化することにより上述の問題点を解決している．このようなルックアップ・テーブルを用いることそれ自体は容易に考え得るところであるが，本論文ではこれを日本全国の道路網に対し実際に適用した場合について，実用的な領域分割数を明らかにすると共に，領域分割の仕方として郡レベルの行政区域に分割すると，テーブル作成に要する時間を多少なりと減少させるのに有利であることを示している．すなわち郡レベルの行政界は一般に河川や山脈等の地形的特性と密接な関係を有していることが多く，このような行政区界を横切る道路は少ないと考えられるためである．本論文ではこのように領域分割の仕方に一つの示唆を与えると共に，ここで採用したルックアップ・テーブル法により探索領域を限定した経路探索法が，やはり最適経路を与えることが保証されている従来のDijkstra法およびA*アルゴリズムと比較し，総所要時間および必要とするRAM容量の面で現用のハードウェアの下では大幅に有利であることを示している．

A Route Finding Method for the Nationwide Road Networks of Japan by Restricting Search Areas through the Use of Look-up Table

Ichiro Iimura† and Masami Kato†

In recent years, a number of onboard navigation systems have become commercially available. Few of them, however, can calculate the optimal route to the destination. Moreover, as the size of the road networks becomes larger, the access and read time of disk storage devices, the capacity of RAM required for calculation and the CPU time increases proportionally, which results in an impractical system. In this paper, the nationwide road networks of Japan are divided into a plurality of closed areas, each of which corresponds to a county. A look-up table showing which closed areas are necessary and sufficient in order to calculate the optimal route was constructed offline. By using this table, the above mentioned problems can be solved. The reason why a county is selected as a closed area is that the border of a county is closely related to the topographical properties such as rivers and mountain ranges. Because only a few roads intersect such borders, the time required to obtain the table can be reduced. This suggests the appropriate division method which should be investigated hereafter. Another feature of the proposed method is that the route obtained is guaranteed to be optimal. This paper shows that the proposed look-up table method is quite effective to reduce the total time and the RAM capacity required to obtain the optimal route compared to the conventional Dijkstra method and *A** algorithm.

1. まえがき

近時，車載ナビゲーション・システムが急速に普及しつつある．しかし，GPS，マップマッチング技術等を用いて自車位置，走行軌跡を表示するものが主体であり，目的点への最短距離あるいは最短時間経路を提供するものは少ない．特に道路ネットワーク・データが膨大な場合に，遠隔2地点が出発地および目的地として指定されると，例えばDijkstra法[1]等従来から良

† 上智大学理工学部電気電子工学科
Department of Electrical and Electronics Engineering, Faculty of Science and Technology, Sophia University

く知られている手法では出発地を中心として四方・八方に探索を行うため，ネットワーク・データの記憶されているディスクあるいはCD-ROM等へのアクセスおよび読み込み時間が長大となるだけでなく，データをロードし探索アルゴリズムを実行するために必要なRAM領域が膨大になるとともに，計算実行時間も実用に耐えなくなる場合がある．このため出発地および目的地が指定されたとき，それらを囲む矩形または楕円の領域に探索領域を限定する方法[2]，A* アルゴリズムを用いる方法[3]，道路網を階層化する方法[4),5)]等が提案されているが，経路が求まらない場合があること，経路が求まっても最適である保証がないこと，計算に時間がかかること等の欠点があった．

本論文では，道路ネットワークのノードは，一般的なネットワークのノードとは異なり，固有の (x, y) 座標を有していることに着目し，膨大な道路ネットワークを複数個の領域に分割し，任意の2つの領域間の最適経路を算出するためにRAMにロードすべき領域を予めすべて求めて，これをルックアップ・テーブルとしている．経路探索実行時に出発点と目的点が指定されると，出発点および目的点がそれぞれ属する領域を求め，これら2つの領域間の最適経路を算出するのに必要とされる領域を前記ルックアップ・テーブルを参照して求め，これら領域に属する道路ネットワーク・データのみをRAMにロードして計算を実行することにより，ディスクへのアクセス・読み込み時間の短縮，必要とされるRAMの容量の減少ならびに計算時間の短縮を図ることが可能となるだけでなく，求められた経路の最適性が保証されることになる[6]．

このようなルックアップ・テーブルを用いる手法それ自体は容易に考え得るところであるが，この手法を膨大なデータ量を有する日本全国の基本道路網（(財)日本デジタル道路地図協会が定義した基本道路網とは，一般都道府県道以上の道路，一般都道府県以外の道路幅員が5.5m以上の道路およびこれらの道路間を連結する連結路により構成される道路網であり，約23万個のノード，約52万本のリンクより成り，これを経路探索に適した形に変換したデータの容量は約8MBを占める）に対して実際に適用した場合，領域分割数を何個位にするのが適当であるかについては従来検討されていなかった．本論文ではテーブルの占める記憶容量ならびにテーブル作成に要する計算時間から現用のハードウェアを用いる場合，分割数を高々500程度にすることが実用的な見地から妥当であることを明らかにしている．

更に本論文においては道路ネットワークを複数個の領域に分割する仕方についても示唆を与えている．すなわち，道路ネットワークを格子状直線により矩形領域に分割することが容易に考えられるが，ここでは郡レベルの行政区域を領域分割の単位として採用した．その理由は郡レベルの行政区界は一般に河川や山脈などの地形的特性と密接な関係を有していることが多く，このような行政区界を横切る道路は少ないと考えられるため，ルックアップ・テーブルの作成に要する時間を減少させるのに有利であることによる．

以下，従来の探索領域限定の手法とその問題点，ここで採用したルックアップ・テーブルによる探索領域の限定手法，道路ネットワークの領域への分割法，データの生成方法，最適経路探索の手順，採用したルックアップ・テーブル法の有効性の評価について詳細に述べる．

2. 従来の探索領域限定法

道路ネットワークにおける経路探索において，道路ネットワーク・データの格納されているディスクへのアクセス・読み込み時間を短縮させ，またロードして探索を実行するためのRAM容量を削減し，経路算出時間を高速化するため，あまり意味がないと考えられるネットワーク・データを経路探索の対象から除外することは従来から行われてきた[2)~5)]．例えば，出発点および目的点が指定されたとき，**図1**（a）および図1（b）に示すようにそれらを囲む矩形または楕円の内部に相当する道路ネットワークのみに探索領域を限定することが考えられる[2)]．ほとんどの場合これで十分な結果が得られるが，例えば図1（a）あるいは図1（b）に示すように出発点と目的点の間の湾や川，山脈等によって経路が見出せない場合が生じ得るし，たとえ得られたとしても最適な経路である保証はない．また日本の道路ネットワークを対象として出発地と目的地が

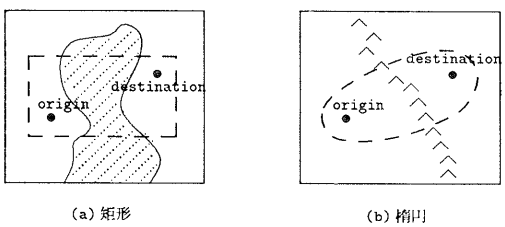


図1 矩形・楕円による探索領域限定
Fig. 1 Rectangular and elliptic search areas.

九州の南端と北海道の北端であるような場合，矩形で探索領域の限定を行うとほとんどすべてのデータが探索領域となってしまい，領域限定の効果は全くない．また楕円で限定を行う場合には楕円内部に属するか否かの判定に時間を要し実用的でない．

探索領域を自動的に限定し，しかも最適な経路を得ることが可能な人工知能的アプローチとして A* アルゴリズムが知られている[3]．良く知られている最適経路探索法である Dijkstra 法では，出発点からの累積コストが最小である未探索のノードから未探索のリンクを探索延伸することにより実行されるのに対し，A* アルゴリズムでは出発点からの累積コストとそのノードから目的点への推定コストの和が最小である未探索のノードから未探索のリンクを探索延伸することにより実行される．

この A* アルゴリズムにおいて推定コストを 0 としたものが通常の Dijkstra 法に相当する．リンクのコストが例えば乱数により与えられているような一般のネットワークにあっては推定コストを与えることは困難であり，従って A* アルゴリズムは適用できない．しかし対象とするネットワークが道路ネットワークのように各ノードが固有の (x, y) 座標を有する場合には，推定コストとしてそのノードと目的点との直線距離に相当するコストを採用すれば求まった経路は最適であることが保証される．この A* アルゴリズムの欠点は，例えば直線距離をコストとする場合には自乗および平方根等の演算を必要とするため算出に時間がかかること，また矩形などで予め探索領域を限定する場合と異なり，探索実行前には探索領域の限定ができず，必要に応じてネットワーク・データを逐次追加読み込みするか，予めすべてのデータを読み込んでおく必要がある．更には日本の道路ネットワークのように細長い形状をしたネットワークを対象とした場合，九州の南端から北海道の北端にいたる経路を探索しようとすると，ほとんどのネットワーク・データが探索の対象となり，探索領域限定の効果は失われてしまう欠点がある．

RAM に読み込むネットワーク・データの量および探索に要する CPU 時間を減少させる他の方法として階層化経路探索が知られている[4),5)]．これは出発地および目的地の近辺でのみデータ量の多い下位の階層の道路ネットワークも使用して最寄りの上位階層の道路ネットワークへの複数個のアクセス・ポイントを見出し，それ以外では上位の道路ネットワークのみを使用して経路探索を行うのであるが，途中で上位の階層から一時的に下位の階層に降りた後，再び上位の階層に戻るといったことが許されない等の道路階層間の遷移に制約があるため，最適な経路が求められる保証がない欠点がある．

3. ルックアップ・テーブルによる探索領域の限定

前述のとおり従来の探索領域限定法には問題点があった．ここで採用した探索領域限定法は，以下に述べるようなルックアップ・テーブルを参照することにより，必要にして十分な領域のみの道路ネットワークをディスクから RAM にロードすることにより，ディスクへのアクセス・読み込み時間を短縮し，必要な RAM 容量を削減し，経路算出時間を高速化するとともに得られた経路の最適性を保証するものである．

道路ネットワークの各データにはそれぞれ固有の (x, y) 座標が対応しているので，この道路ネットワークを複数個の領域に容易に分割することができる．例えば，**図 2** に示すような N 個の領域に道路ネットワークを分割した場合を考える．このとき領域を形成する閉曲線と道路ネットワークのリンクとの交点ノードを境界ノードと呼ぶことにする．例えば領域 $\#i$ に属する出発点 O から，領域 $\#j$ に属する目的点 D に至る最適経路を探索することを考える．領域 $\#i, \#j$ の境界ノード数をそれぞれ k_i, k_j とし，領域 $\#i, \#j$ の第 i' および第 j' 境界ノードをそれぞれ $n(i, i')$, $n(j, j')$ とする．出発点 O と目的点 D が異なる領域に属する

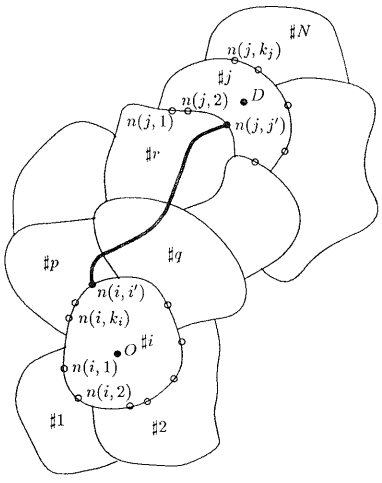


図 2 複数個の領域への分割
Fig. 2 Division into a plurality of areas.

場合，すなわち $i \neq j$ のとき，出発点 O から目的点 D に至る最適経路は領域 $\#i$ の境界ノード $n(i, i')$, $(i'=1 \sim k_i)$ のいずれかから出て，領域 $\#j$ の境界ノード $n(j, j')$, $(j'=1 \sim k_j)$ のいずれかから入ることは明らかである．例えば境界ノード $n(i, i')$ から $n(j, j')$ に至る最適経路が領域 $\#p$, $\#q$, $\#r$ を通るならば，これらの領域は領域 $\#i$ に属する出発点 O から領域 $\#j$ に属する目的点 D に至る最適経路を探索する場合に，領域 $\#p$, $\#q$, $\#r$ に属する道路ネットワーク・データは必須であることがわかる．このようにして領域 $\#i$ の k_i 個の境界ノード $n(i, i')$, $(i'=1 \sim k_i)$ をそれぞれ出発点，領域 $\#j$ の k_j 個の境界ノード $n(j, j')$, $(j'=1 \sim k_j)$ をそれぞれ目的点とする $k_i \times k_j$ 通りの最適経路を算出し，それらすべての経路が通過する領域の集合が，領域 $\#i$ に属する任意の出発点 O から領域 $\#j$ に属する任意の目的点 D に至る最適経路を算出する場合に必要にして十分な領域の集合となる．道路ネットワークが双方向性のときは，出発点が領域 $\#i$，目的点が領域 $\#j$ に属する場合に必要にして十分な領域の集合と，出発点が領域 $\#j$，目的点が領域 $\#i$ に属する場合に必要にして十分な領域の集合は同一であるが，双方向性が成り立たないときは2つの集合は異なるがその差異は僅少であると考えられるので記憶容量を削減するため，ここでは両集合の和集合を用いることにした．

また出発点 O と目的点 D が同一の閉集合 $\#i$ に属する場合にも同様にして最適経路を求めるのに必要にして十分な領域の集合が求められるが，**図3**に示すように領域 $\#i$ のみのデータでは十分でない場合が多いことに注意する必要がある．

このようにして**表1**に示すような領域 $\#i$ と領域 $\#j$ の間の最適経路を探索するのに必要にして十分な領域の集合を表現するテーブル（これをルックアップ・テーブルと呼ぶ）が形成される．

このようなルックアップ・テーブルが与えられれば出発点・目的点が与えられたとき，それぞれの属する領域番号に相応するルックアップ・テーブルの所定の1行分のデータをランダム・アクセスで読み込むことにより，RAM にロードすべき道路ネットワーク・データを限定することができ，これらの道路ネットワーク・データのみを用いて算出した経路は最適であることは明らかである．

4. 道路ネットワークの分割法

表1に示すルックアップ・テーブルのサイズは行方向 N，列方向 $N(N+1)/2(=\sum_{i=1}^{N} i)$ であり，テーブルの0または1を1ビットで表現することにすると

$$\frac{1}{2}N^2(N+1) \text{ ビット} \qquad (1)$$

の記憶容量を必要とすることになる．領域分割数 N とテーブルのファイル容量の関係を**図4**に示す．

一般に分割数 N を大きくすれば探索のために RAM にロードすべきデータ量は減り，読み込みアクセス時間も減少するが，テーブル・サイズおよびテーブル作成時間の面で不利となるために適当な N を選定する必要がある．$N=1$, すなわち分割を行わない場合にはテーブルは必要ないのでファイル容量は0であるが，当然領域は限定されない．次に例えば都道府県を単位として領域分割した場合には $N=47$ であるのでテーブルのファイル容量は約 6.6 KB と極めて小容量で済むが，後述するように限定される領域単位が大きす

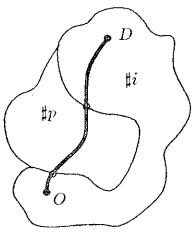


図 3 特殊な例
Fig. 3 A particular case.

表 1 ルックアップ・テーブル
Table 1 A look-up table.

出発領域	目的領域	$\#i \leftrightarrow \#j$ 最適経路探索に必要十分な閉領域の集合を表すテーブル								
		$\#1$	$\#2$	$\#3$	…	…	…	…	…	$\#N$
$\#1$	$\#1$	1	*	*	…	…	…	…	…	*
$\#1$	$\#2$	1	1	*	…	…	…	…	…	*
$\#1$	$\#3$	1	*	1	…	…	…	…	…	*
⋮	⋮	⋮								
$\#1$	$\#N$	1	*	*	…	…	…	…	…	1
$\#2$	$\#2$	*	1	*	…	…	…	…	…	*
$\#2$	$\#3$	*	1	1	…	…	…	…	…	*
⋮	⋮		⋮							
$\#2$	$\#N$	*	1	*	…	…	…	…	…	1
$\#3$	$\#3$	*	*	1	…	…	…	…	…	*
⋮	⋮			⋮						
$\#3$	$\#N$	*	*	1	…	…	…	…	…	1
≀	≀									
$\#N$	$\#N$	*	*	*	…	…	…	…	…	1

1：必要
0：不要
*：0または1

ぎ，領域限定の効果が発揮できない．領域分割の単位を後に詳述する郡レベルの行政区域とすると分割数Nは513となり，テーブルのサイズが約8.5MBとかなり大きいが，経路探索用データのサイズが前述のごとく約8MBであること，得られた経路を地図上に表示するためのノードおよび補間点の(x, y)座標データが約24MB，また水系，鉄道，行政界，地名などの背景データが約134MBであることを考えると，現用のハードディスクあるいはCD-ROMの記憶容量を考慮に入れると妥当な値と考えられる．更に分割の単位を市区町村とすると分割数Nは3,372となりテーブル・サイズは約2.4GBを占め，最早現用の記憶装置では非現実的となる．

ちなみにここで使用した(財)日本デジタル道路地図協会の日本全国の道路ネットワークはJISで規定された約10km四方の第2次地域区画（以下では2次メッシュと呼ぶ）で分割されているが，道路ネットワークの存在する2次メッシュ数は4,353であり，この2次メッシュを分割の単位としてテーブルを作るとそのサイズは約5.2GBとなりやはり実用的でない．

分割の極限状態は領域がただ1つのノードを含む場合で，これは任意の2つのノード間の最適経路を予め求めてテーブル化することに相当するが，$N=23$万とするとテーブル・サイズは天文学的数字となり問題外である．

次にルックアップ・テーブル作成に要する計算時間について考察する．テーブルを作成するためには各境界ノードから他のすべての境界ノードに至る最適経路をオフラインで求めることになるが，1つの境界ノードを出発地として他のすべての境界ノードに至る最適経路は一回の処理でまとめて求めることができ，その計算時間は日本全国の道路ネットワークのノードとリンクをほとんどすべて既探索とする必要があるのでほぼ一定と考えてよい．それゆえ，テーブルを求めるための計算時間は境界ノードの数 $K=\sum_{i=1}^{N} k_i/2$（1つの境界ノードは2つの領域が共有するので2で割っている）に比例することになる．従って境界ノード数を減少させることがテーブル作成時間の観点からは望ましい．境界ノード数Kは分割数Nが増えると一般に増加し，リンクが縦・横方向に一様密度で存在する場合格子状に分割すると，分割数をn倍すると境界ノード数は$\sqrt{n}$倍になることが容易に示される．実際の日本全国の基本道路網のリンクの分布は一様でないが，これを格子状に分割した場合の境界ノード数Kをいくつかの分割数Nに対して求めた結果を**図5**に示す．これは次のようにして求めた．すなわち，筆者らが使用した道路ネットワーク・データは前述のごとく約10km四方の2次メッシュ4,353枚より成っているが，これを縦・横方向に整数枚並べて格子状メッシュを構成したときの境界ノードを実際に計算機で計数したものである．この境界ノード数データは使用した(財)日本デジタル道路地図協会のネットワーク・データが2次メッシュの区画辺交点ノードを有しているために容易に求めることができた．

このように道路ネットワークを格子状に分割するこ

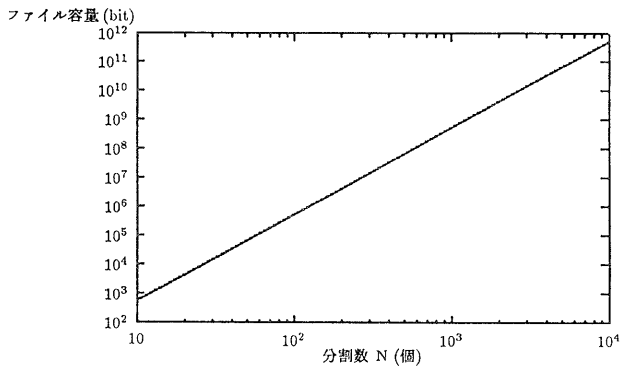


図4 分割数Nとファイル容量
Fig. 4 No. of division N vs. file capacity.

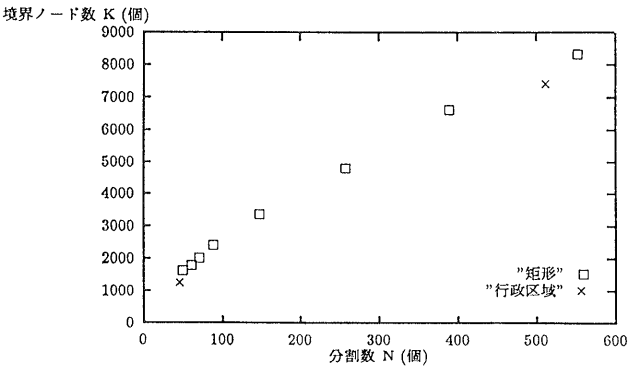


図5 分割数Nと境界ノード数K
Fig. 5 No. of division N vs. no. of boundary nodes K.

とは容易であるが，筆者らは同じ数に分割したときに境界ノードの数を少しでも減少させ得る分割法としてどのような方法が考えられるかについて検討を行った．その結果，一般に河川や山脈を横切る道路は少ないことに着目し，その地形的特性と密接な関係を有していることが多い行政区域を分割単位として採用すると境界ノードを減らし得るものと予想した．

図6に行政区域として都道府県を採用し，$N=47$ に分割した場合の各都道府県の境界ノード数をJISの都道府県行政区域コード（例えば北海道01，青森県02，沖縄県47）の関数として示す．これより東京都が最大の境界ノード数160を有し，境界ノード総数Kは1,261となり，図5に示す前述の格子状分割した場合の境界ノード数に比べ減少傾向にあることが分かる．なおこのような行政区域単位への道路ネットワークの分割は筆者らが使用した(財)日本デジタル道路地図協会のデータには行政区域の変化点に相当するノードが存在し，またリンクの属性として都道府県・市区町村コードが付与されているので容易に行うことができた．

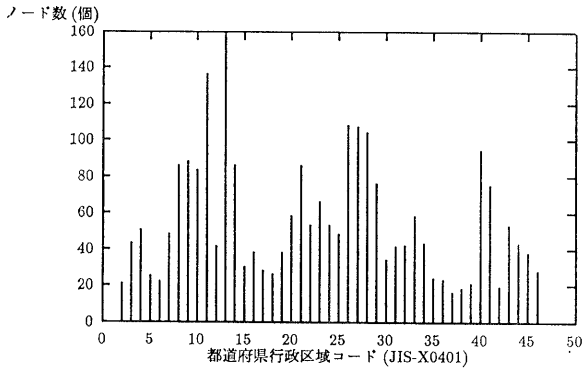


図6 都道府県別境界ノード数
Fig. 6 No. of boundary nodes as a function of area code.

図6から分かるとおり，行政区域コード06に対応する山形県の境界ノード総数は23であるが，**図7**はこの山形県の境界ノードの実際の位置を示すものである．

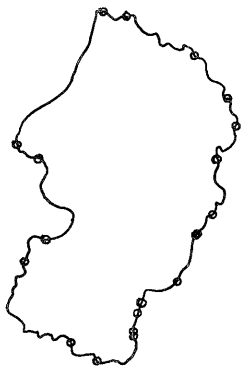


図7 山形県の境界ノード
Fig. 7 Boundary nodes of Yamagata prefecture.

このように都道府県を単位とすると境界ノード数を削減できることは，身近な例で言えば東京都の境界の一部を成す多摩川・江戸川・荒川などには数えるほどの橋しかないこと，東京都北西部の埼玉県・山梨県との境界はほぼ山の尾根であってほとんど道がないことなど，地形的特性を近似的に取り込んでいるためと考えられる．しかし都道府県を分割の単位とすると限定される範囲が大きすぎ，九州の南端から北海道の北端までの経路を求めるような場合にはほとんどすべての領域が探索領域となるため，ルックアップ・テーブルを使用して領域限定を行う効果はほとんど失われる．

前述のごとく郡レベルの行政区域を単位として513に分割すると，テーブル・サイズは約8.5MBと実用的な値となる．しかし近年，行政区域の合併・昇格により市政を布くところが多く，郡という単位は形骸化している．そこで筆者らは昭和23年当時の地図[7]を参照して，現在市政を布いているところも手作業で当時の郡（北海道の場合は支庁）に編入して全国を513個の郡レベルに相当する領域に分割した．例えば“東京都”は“23区”，“北多摩郡”，“西多摩郡”，“南多摩郡”の4つの領域に分割した．その結果“23区”が513の郡レベルの領域の中で最大の境界ノード数118を有し，境界ノード総数は7,425となり，図5から分かるように格子状分割した場合よりも境界ノード数は多少ではあるが減少していることが分かる．減少の度合いは期待をはるかに下まわり，日本全国の道路ネットワークを郡レベルに相当する領域に分割するのに要した時間および労力と比べると極めて不満足ではあるが，これはここで採用した郡レベルの行政区界が必ずしも地形的特性と一致していないためと考えられる．しかしながら今後更に検討すべき望ましい領域分割の仕方に1つの展望を与えるものと考えられる．また箱根・碓井・新居といったかつての関所のような必ず通らなければならないようなボトルネックの点を探すこと等と併せて更に詳細な検討を行う必要があるものと考えられる．

ところでこのように513の郡レベルに分割した場合，境界ノード数は7,425個であるから，ルックアップ・テーブルを作成するには7,425回出発点をかえて他の7,424点に至る最適経路を求め，求められた経路を構成するリンクがどの領域に属するかを調べる操作が必要とされる．このようにすべての境界ノード間の最適経路を求め，その最適経路が通る領域を調べて作成したルックアップ・テーブルを使うのであって，最適経路そのものを使うのではないことに注意されたい．これはワークステーション SPARCstation 10 を用いて約80時間を要したが，十分実用的な時間であると考えられる．

以上の考察の結果，ルックアップ・テーブルの容量，その作成時間，探索領域限定の効果の観点から郡レベルの513の領域に分割することは実用上適当であると考えられ，以下ではこの領域分割を採用したルックアップ・テーブルを用いた経路探索法について検討を加えることにする．

5. 経路探索の手順

ルックアップ・テーブルにより探索領域を限定して最適経路の探索を行う処理の手順は次のとおりである．

ステップ 1：出発点，目的点がそれぞれ属する領域番号 $\#i$, $\#j$ を求める．

ステップ 2：($i \leq j$ のとき) は表2に示すルックアップ・テーブルの $(\#i, \#j)$ に相当する一行を，($i > j$ のとき) は $(\#j, \#i)$ に相当する一行をランダム・アクセスによって RAM 中に読み込む．

ステップ 3：読み込まれた領域集合限定情報に応じて道路ネットワーク・データをディスクから逐次読み込み，各領域ごとに1から順番に付けられたノードおよびリンク番号を絶対的な連続番号に変換する．その際図8に示すように，相応する境界ノード間をコスト0の仮想リンクにより相互接続し，これにも絶対的な連続番号を付与する．

ステップ 4：得られた必要にして十分なネットワーク・データに対して Dijkstra 法を適用して最適経路を探索する．

6. 提案した手法の評価

4章で述べた日本全国の基本道路網を513の郡レベルの領域に分割し，ルックアップ・テーブルを用いて探索領域を限定することにより経路探索を行う手法の利点を明らかにするために，やはり最適経路が求められることが保証されている通常の Dijkstra 法および A* アルゴリズムを適用した場合との比較を行った．比較を同一条件の下で行うため，Dijkstra 法および A* アルゴリズムを適用する場合のデータも，513の郡レベルの領域に分割した道路ネットワーク・データを用い，出発点から目的点への最適経路が見出されるまで，必要に応じて郡レベルの領域の道路ネットワーク・データを逐次ディスクにアクセスして読み込み，ノードおよびリンクに絶対的な連続番号を付与して1つの大きな道路ネットワーク・データを作るものとした．

これら3つの手法の処理時間面での優劣は出発点および目的点が指定された後，ディスクから道路ネットワーク・データを読み込んで，探索アルゴリズムを実行して最適経路が求められるまでの総所要時間で評価される．実際のシステムでは通常求められた経路を背景地図と共にディスプレイに表示する必要があり，そのためにはノードやリンクの補間点の (x, y) 座標など場合によっては多量のデータを読み込む必要があり，時間を要することになるが，いずれの手法に対しても同様であるのでここでは考えないこととする．ところで評価の基準となる前記総所要時間は，必要とされる領域ごとにディスクにアクセスしてデータを読み込んでノードおよびリンクに絶対的な連続番号を付与するという前処理を行って1つの大きな道路ネットワーク・データを作る時間（これをディスクアクセス読み込み時間および前処理時間と呼ぶ）と，得られた

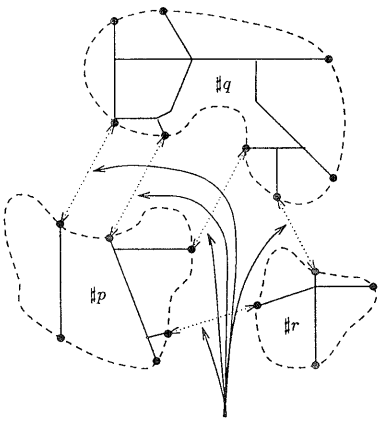


図 8 仮想リンクの導入
Fig. 8 Introduction of virtual links.

道路ネットワーク・データに対し探索アルゴリズムを実行して最適経路を求める時間（これを探索アルゴリズム実行時間と呼ぶ）に分けることができる．ただしルックアップ・テーブル法ではテーブルにランダム・アクセスして513ビットのデータを読み込む時間をディスクアクセス読み込み時間および前処理時間に含めているが，これはほとんど無視し得る時間である．

前記ディスクアクセス読み込み時間および前処理時間は筆者らが使用した計算機ではそれぞれの内訳を測定することができなかったので，ディスクアクセス時間に関係すると考えられる読み込んだ領域の数，およびディスク読み込み時間と前処理時間ならびに必要とされるRAM領域に関係すると考えられる読み込まれたリンク本数を参考のために計数した．また探索アルゴリズム実行時間と関係する既探索とされたリンク本数も参考のために計数した．

以下で出発地および目的地の3通りの組み合わせに対して，リンクのコストを距離とすることは実用的に余り意味が無いと考えられるのでリンクのコストを時間として最適経路の探索を行った場合について3つの手法の評価を行った結果を**表2**にまとめて示す．この場合道路種別による平均速度は次のように仮定した．高速自動車国道＝100 km/h，都市高速道路＝70 km/h，一般国道＝55 km/h，主要地方道＝50 km/h，一般都道府県道＝40 km/h，一般都道府県道以外の道路幅員が5.5 m以上の道路およびこれらの道路間を連結する連絡路＝40 km/h，フェリー＝8 km/h．なおA^*アルゴリズムを適用する場合の推定コストとして直線距離に相当する高速自動車国道の時間を使用すれば良いことは明らかである．ここで使用した計算機はSPARC station 370である．

［例1］ 出発地＝能登半島先端
目的地＝伊豆半島先端

表2に示す結果より次のようなことが分かる．総所要時間の点ではここで採用したルックアップ・テーブル法が7.13秒であるのに対し，Dijkstra法はその約9.5倍の67.89秒，A*アルゴリズムはその約7.6倍の54.02秒を要しており，この場合本手法が明らかに有利である．その内訳であるディスクアクセス読み込み時間および前処理時間に関しては，本手法は38の領域の54,804本のリンクを読み込んで1つの道路ネットワークを形成すればよいので5.77秒で済んでいるのに対し，Dijkstra法では341の領域の372,448本のリンクを読み込まねばならないのでその約10.5倍の60.48秒を要し，A*アルゴリズムではDijkstra法と比べれば減少しているがそれでも193の領域の252,948本のリンクを読み込まねばならないのでその約6.0倍の34.87秒を要している．

図9〜図11にそれぞれ本手法，Dijkstra法，A*アルゴリズムを用いた場合に読み込まれたリンクと求められた最適経路を示す．これよりDijkstra法およびA*アルゴリズムが最終的には不要となる大量の道路ネットワークデータをいかに読み込んでいるかが分かる．

一方探索アルゴリズムの実行時間は本手法が1.36秒であるのに対し，Dijkstra法では約5.4倍の7.41秒を要している．本手法も探索アルゴリズムとしてはDijkstra法を用いているが探索の対象となるネットワークのリンク本数が54,804本と少ないため，既探索とされたリンク本数は54,465本であるのに対し，探索領域を限定しないでDijkstra法を適用した場合には探索の対象となったネットワークのリンク本数は

表2 評価結果
Table 2 The experimental results.

	探索手法	総所要時間 (sec)	ディスクアクセス読み込み時間及び前処理時間 (sec)	探索アルゴリズム実行時間 (sec)	読み込まれたリンク本数	既探索とされたリンク本数	読み込んだ領域数
例1	本手法	7.13	5.77	1.36	54,804	54,465	38
	Dijkstra法	67.89	60.48	7.41	372,448	353,997	341
	A^*アルゴリズム	54.02	34.87	19.15	252,948	230,924	193
例2	本手法	3.72	3.00	0.72	25,556	23,254	27
	Dijkstra法	39.21	35.49	3.72	188,444	177,995	212
	A^*アルゴリズム	28.16	20.69	7.47	108,270	92,126	140
例3	本手法	29.65	24.37	5.28	209,522	208,510	128
	Dijkstra法	126.99	114.27	12.72	530,358	529,576	503
	A^*アルゴリズム	159.67	114.27	45.40	530,358	528,688	503

372,448 本で，そのうち既探索とされたリンク本数は 353,997 本と本手法の約 6.5 倍となっており，そのため探索アルゴリズムの実行時間も約 5.4 倍の時間を要している．なお A*アルゴリズムは探索領域を自動的に制限する効果があるが，推定コストの計算に時間がかかるため，Dijkstra 法の探索アルゴリズム時間が 7.41 秒であるのに対し，A* アルゴリズムでは 19.15 秒と約 2.6 倍の時間を要し，ディスクアクセス読み込み時間および前処理時間が削減された利点を相当程度相殺している．

［例 2］ 出発地＝山口県西部
目的地＝高知県中央

図 12 に本手法を用いた場合に読み込まれたリンクと求められた最適経路を示す．表 2 に示す結果は例 1 の場合とほとんど同様な傾向を示しており，この場合にも本手法の有効性は明らかである．

［例 3］ 出発地＝九州佐多岬
目的地＝北海道知床岬

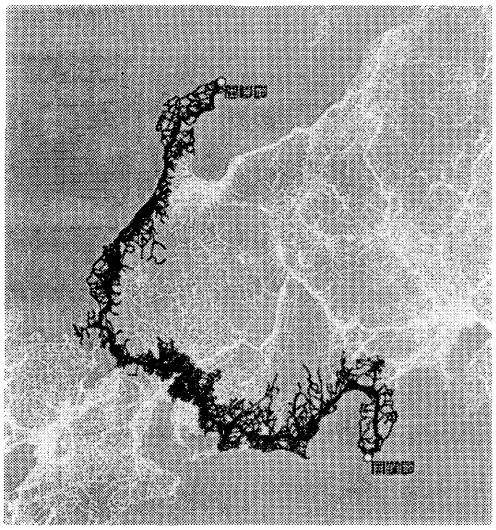


図 9 例 1（本手法）
Fig. 9 Example 1 (proposed method).

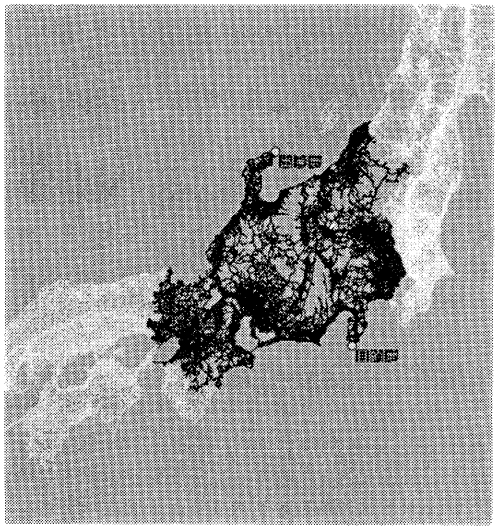


図 11 例 1（A* アルゴリズム）
Fig. 11 Example 1 (*A** algorithm).

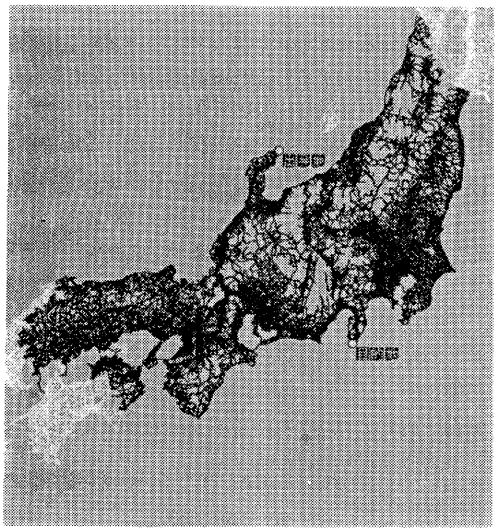


図 10 例 1（Dijkstra 法）
Fig. 10 Example 1 (Dijkstra method).

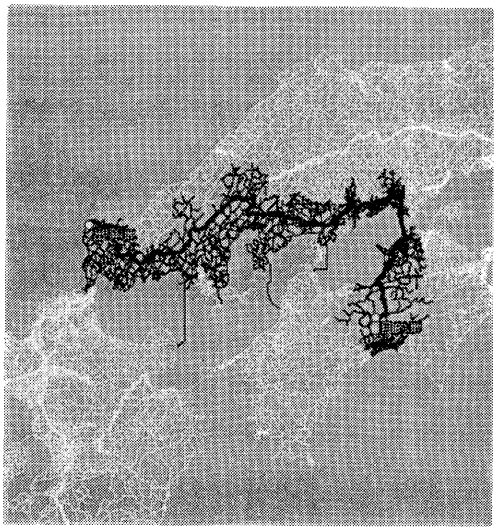


図 12 例 2（本手法）
Fig. 12 Example 2 (proposed method).

図 13 に本手法を用いた場合に読み込まれたリンクと求められた最適経路を，表2に評価結果を示す．この例はいずれの手法を使用しても最も時間のかかるケースに相当し，最悪条件下での各手法の評価を与えるものである．総所要時間では本手法 29.65 秒，Dijkstra 法 126.99 秒，A* アルゴリズム 159.67 秒と本手法以外では2分～3分というオーダになる．この場合，本手法では 128 の領域の 209,522 本のリンクに対するディスクアクセス読み込み時間および前処理時間は 24.37 秒であるのに対し，Dijkstra 法，A* アルゴリズムとも沖縄を除く日本のほぼすべてに相当する 503 の領域の 530,358 本のリンクに対するディスクアクセス読み込み時間および前処理時間は約 4.7 倍の 114.27 秒となっている．一方，本手法の探索アルゴリズム実行時間は 5.28 秒であるのに対し，探索領域を限定しない通常の Dijkstra 法の探索アルゴリズムは 12.72 秒と 2.4 倍にすぎないが，ディスクアクセス読み込み時間および前処理時間が総所要時間に対して支配的な大きさを占めるため本手法の優位性は明らかである．またこの場合は A* アルゴリズムの探索領域の自動制限は全くといって良いほど機能していないだけでなく，推定コストの計算のため，総所要時間の面で通常の Dijkstra 法よりも劣る結果を与えることになる．なお探索実行のために必要なRAMの最大のワーキングエリアは読み込まれたリンク本数に比例すると考えられるが，本手法では読み込まれたリンク本数が

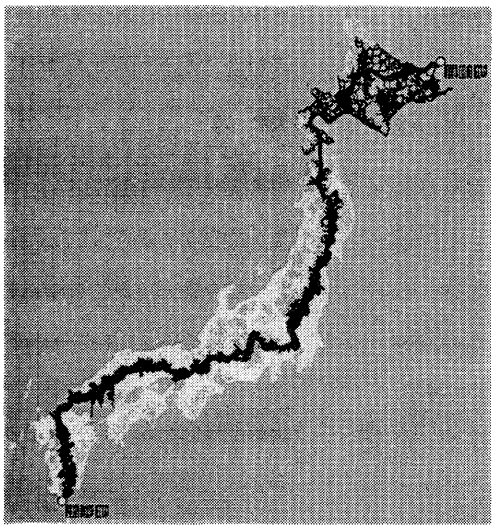


図 13 例3（本手法）
Fig. 13 Example 3 (proposed method).

209,522 本であるのに対し，Dijkstra 法，A* アルゴリズムでは 530,358 本であるので本手法の必要とするRAM エリアは他の方法の約 40% で済むことになる．

以上，本手法は総所要時間，その内訳であるディスクアクセス読み込み時間および前処理時間ならびに探索アルゴリズム実行時間，そして必要とするRAMエリアのいずれの面でも優位であり，実用上極めて有効であると言える．

7. むすび

本論文では日本全国の基本道路網を複数個の領域に分割し，ルックアップ・テーブルを用いて探索領域を限定する経路探索手法について考察を加え，分割数としては道路ネットワーク・データのサイズおよび現用のハードディスクの記憶容量から 500 程度が限度であることを示し，領域分割の仕方としては 513 の郡レベルの行政区域を領域の単位として採用した．その結果得られた境界ノード数の削減効果は期待を下まわるものであったが，格子状分割よりも境界ノード数を減少させるのに有効であると思われる領域分割の仕方に1つの方向性を与えるものと考えられる．そしてここで提案した手法を，やはり最適経路が得られることが保証されている Dijkstra 法および A* アルゴリズムと比較し，現用のハードウェアの下では総所要時間，その内訳であるディスクアクセス読み込み時間および前処理時間ならびに探索アルゴリズム実行時間，そして必要とされるRAM領域を削減するのに大幅に有利であることを示した．

本手法の問題点はルックアップ・テーブルの作成にかなりの時間を要すること，渋滞や事故などの時間変動を考慮に入れられないこと，新しい道路ができた時にはテーブルを作り直さなければならないこと等があげられる．ここでは格子状分割よりも多少ではあるが境界ノード数を減少させる郡レベルの行政区域への分割を行ったが，さらに地形的特性をより良く表現する望ましい領域分割法について今後検討を行う必要がある．更には検索が容易で情報圧縮が可能なルックアップ・テーブルの管理方式を考案することも課題の1つである．

謝辞 最後に有益な御討論戴いた本学マルチメディア・ラボの諸氏に謝意を表する．

参考文献

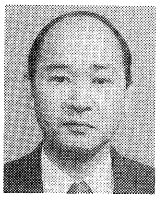
1) 伊理正夫，中森真理雄：算法の最近の進歩，信

学誌, Vol. 58, No. 4, pp. 433-445 (1975).
2) 小林祥延, 平野和夫, 出川裕久, 橋本武夫, 名倉充彦：推奨経路表示機能付きナビゲーションシステム, 住友電気, 第141号, pp. 151-160 (1992).
3) Nilsson, N. J. (著), 白井良明, 辻井潤一, 佐藤泰介(訳)：人工知能の原理, 日本コンピュータ協会 (1983).
4) 加藤誠巳, 大西啓介：階層化されたディジタル地図データベースに基づく都心部自動車用経路案内システム, 信学技報, DE 89-24 (1989).
5) 丹羽寿男, 吉田雄二, 福村晃夫：道路網の階層的表現にもとづく経路探索アルゴリズムと地図情報システムへの応用, 情報処理学会論文誌, Vol. 31, No. 5, pp. 659-666 (1990).
6) 飯村伊智郎, 加藤誠巳：地形的特性により探索領域を限定した日本全国道路網における経路探索手法, 情報処理学会情報システム研究会資料, IS-48-3 (1994. 3).
7) 日本地方行政研究會編纂：全國市町村便攬 附・分縣地圖, 日本観光出版株式會社 (1948).

(平成6年7月12日受付)
(平成6年9月 6 日採録)

飯村伊智郎 (学生会員)

昭和 44 年生. 平成4年上智大学理工学部電気電子工学科卒業. 平成6年同大学大学院博士前期課程修了. 同年(株)日立製作所入社. 以来, 日立研究所にて, 3次元コンピュータ・グラフィックス, マルチメディア処理の研究開発に従事.



加藤 誠巳 (正会員)

昭和17年生. 昭和40年東京大学工学部電子工学科卒業. 昭和 42 年同大学院工学系研究科電子工学専門課程修士課程修了. 同年日本電信電話公社電気通信研究所入所. 東京大学工学部研究生を経て昭和 48 年東京大学大学院工学系研究科電子工学専門課程博士課程修了. 工学博士. 昭和 48 年上智大学理工学部電気電子工学科講師, 昭和49年同助教授, 昭和57年同教授, 現在に至る. 都市交通情報提供システム, コンピュータ・グラフィックス, ニューラルネット, 画像処理, 音声信号処理, 人工知能等の研究に従事. IEEE, 電子情報通信学会, 日本オペレーションズ・リサーチ学会, 日本音響学会, 日本シミュレーション学会, 電気学会各会員.