

ネットワーク分割による交通量配分簡略化法の数値的検討

京都大学工学部	正 員	飯田恭敬
京都大学工学部	正 員	朝倉康夫
京都大学大学院	学生員	廣川誠一
京都大学工学部	学生員	○鷹尾和享

1. 序

一般に、ネットワークの交通量配分計算には膨大な計算時間を要し、その大部分は、最短経路探索に費やされる。その計算時間は、一般に（セントロイド数）×（ノード数の2乗）に比例するとされている（井上（1975）など）。そこで、本研究では、ネットワークをいくつかのブロックに分割し、階層的に最短経路探索を行うことにより、分割配分法（IA法：Incremental Assignment）の計算時間の短縮を図る方法について考察した。ネットワークを分割すると、個々の最短経路探索についてのネットワーク規模が小さくなり、それが2乗で効いてくるので、全体の計算時間は短縮できることが予想される。

なお、以下の文中でネットワークの分割数（ブロック数）のことを「分割数」、OD表の分割数のことを「繰り返し回数」と呼ぶことにする。

2. ブロック分割の考え方と計算手順

ブロック分割による計算法の概略は図1の通りである。ブロックに分割されたネットワークを下位ネットワーク、仮リンクによる全域のネットワークを上位ネットワークとすると、両者の関係は、下位ネットワークから上位ネットワークへ仮リンクの所要時間が与えられ、上位ネットワークから下位ネットワークへODフローが与えられるというものである。数値計算は、ノード数が441、リンク数1680、セントロイド数25の仮想的なネットワークを用いて行った。計算に当たっては、分割数と計算時間の関係を調べるという目的のために、ブロックの分割を自動的に行うようなプログラムを作成し、分割数を自由に変えられるようにした。

原則的にこの計算では、ネットワークを分割して求めた最短経路は、ネットワークを分割しないで求めた最短経路と全く同じ経路になる。

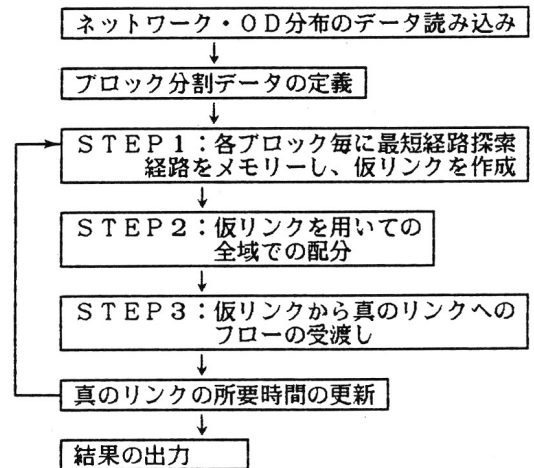


図1 ブロック分割による計算のフローチャート

表1 計算時間・メモリーサイズ

分割数	CPU TIME (ミリ秒)	内訳 (ミリ秒)			TOTAL CPU (ミリ秒)	メモリー (KB)
		STEP1	STEP2	STEP3		
従前	1598	—	—	—	16,388	58
4	818	761	52	5	8,395	152
9	534	400	129	5	5,562	194
16	414	245	164	4	4,363	203
25	511	207	300	4	5,332	230
36	565	180	381	4	5,876	270
49	666	170	493	4	6,888	311

CPU TIME : 繰り返し部分の1回あたりの平均
TOTAL CPU : コンパイルを始めてから繰り返しを10回
行い、結果を出力するまでの全計算時間

3. 計算結果

分割数を4分割から49分割まで変化させた時の計算結果を、表1に示す。繰り返し回数は、全て10回とした。これらの表から次のことがわかる。

- ①ステップ1では、分割数を増やすにしたがって計算時間は減少している。
- ②ステップ2では逆に、分割数を増やすにしたがって計算時間が長くなっている。
- ③一旦経路を記憶するので、ステップ3での配分計算にはほとんど時間はかかっていない。
- ④その結果、全体の計算時間は16分割で最小となっている。したがって、このネットワークの最適分割数は16であるといえる。その時の計算時間は従前の方法の約4分の1にまで短縮されている。
- ⑤必要なメモリーサイズは、分割数を増やすにつれて次第に増加している。

4. 考察

ダイクストラ(Dijkstra)法による最短経路探索の計算手順は図2の通りである。計算時間を検討するために、(ア)～(ウ)までの各ステップで繰り返し回数1回あたりの、抽出しなければならないノードまたはリンクの総個数を考えて、その計算回数を数え、各ステップの計算時間を推定してみると、おおむね

$$(ア) : (イ) : (ウ) = 1 : 4 : \text{ノード数} \quad (1)$$

となる。したがって、最短経路探索の計算時間のうちのほとんどが(ウ)のステップで費やされていることが推定できる。いま、(ア)と(イ)を無視して(ウ)に着目すると、繰り返し1回あたりの回数は、一般に言われているように、

$$(\text{セントロイド数}) \times (\text{ノード数})^2 \quad (2)$$

に比例していることが確認できる。したがって、

- ①ステップ1では、ブロックに分割すると、ノード数の減少が2乗で効いてくるので、分割数を増やすほど計算時間は減少していく。
- ②ステップ2では、分割数を増やすと境界ノードの数が増加し、それが2乗で効いてくるので、分割数を増やして行くと、計算時間は加速度的に増加していく。

①、②はトレードオフの関係にあるので、全体の計算時間はある分割数で最小となる。

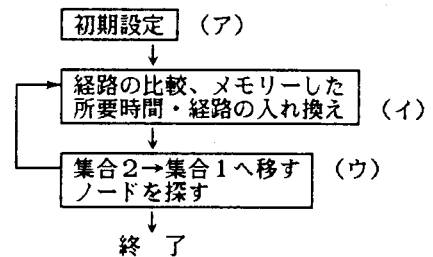


図2 ダイクストラ法のフローチャート

5. 結論

- ①ブロックに分割して階層的に経路探索を行うと、従前のIA法より計算時間が短縮され、ある分割数で計算時間は最小となる。
- ②配分計算の大部分は最短経路探索に費やされており、その計算時間は、 $(\text{セントロイド数}) \times (\text{ノード数})^2$ に比例していることを確認した。
- ③所要時間の等しい最短経路が複数存在すると、配分結果は必ずしも一意的には定まらない。
- ④分割数を増やすと境界ノードが増加し、計算時間の短縮・メモリーサイズにとって好ましくない。これを改良するには、境界ノードを束ねる等の工夫をするといふ。
- ⑤メモリーサイズは従前の方法の数倍を要することとなったが、ブロック単位で別々に計算を行うという工夫をすることによって、従前の方法よりメモリーサイズを小さくすることも可能である。
- ⑥最短経路探索の計算時間が式(2)のようになるのは、手法の改良が不十分なためであり、改良次第では計算時間は大幅に短縮できる。

(参考文献) 井上博司(1975): 道路網における等時間原則による交通量配分に関する基礎的研究、

京都大学学位論文、p.110

土木学会編: 交通需要予測ハンドブック、pp.100-102