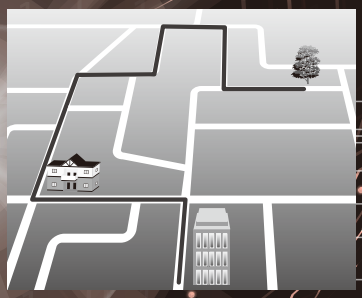
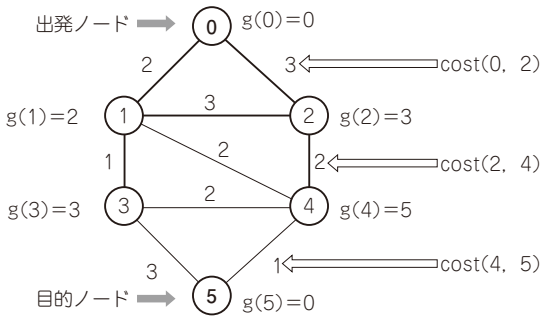
身の回りにある現象や問題をモデル化して解く

数理最適化プログラミング



第2回　最短経路問題を解くダイクストラ法

牧野 浩二



g(*n*)：出発ノードからの距離→更新される値
cost(*n*, *m*)：要素間の距離→あらかじめ決められている値

図1　最短経路を見つけるダイクストラ法

本連載では，現実の問題を数理モデルに当てはめ，数多くの選択肢の中から最も望ましい組み合わせを見つけ出す数理最適化問題を扱います．

今回は数理最適化の問題として最も短い経路を見つけ出す問題でよく用いられる方法「ダイクストラ法」を紹介します．

● 数理モデルや数理最適化は役立つ

数理モデルとは，私たちの身の回りにある現象や問題を数式やアルゴリズム（数理モデル）に置き換えたものです．例えば，うわさ話やインフルエンザなどの病気が広がる様子をモデル化して予測するといったことも数理モデルの1つです．

数理最適化問題は，ある駅からある駅へ電車を乗り継いでいくときの最短となる経路を求めるときにも用いられます．これは簡単に見えますが，駅の数が増えると選択肢が増えるため全部の組み合わせを調べることができなくなります．そこで，いろいろな方法が提案されています．

● 日常で最短経路問題を考える場面

最短経路問題は**図1**に示すように，ノードを経由地として，出発ノードから目的ノードへ移動するときの最短経路を見つけ出す問題です．

このとき，エッジ（ノード間をつなぐ道）にはそれぞれ距離や時間などに相当する値が設定されています．例えばこれは，駅をノードして駅と駅の間の時間をエッジの値とすることで最短経路を求めることにも応用できます．

ダイクストラ法の原理

最短経路を見つける方法はいろいろなものが考えられていますが，今回はダイクストラ法というエッジに与えられた値をコストとして扱うことで最適な経路を見つける方法を紹介します．

なお，ダイクストラ法は全ての組み合わせを試すため最適な経路を見つけることができますが，ダイクストラ法の計算量はエッジの数の2乗に比例する計算量が必要です．そのため，問題のサイズが大きくなると，必要とされる計算量がとても多くなります．計算時間を短くする方法は今後紹介します．

ダイクストラ法のアルゴリズム

各ノードには出発したノードからのそのノードに到達するまでにたどったコストの合計値として，ノードの値［g(n)］が設定されています（**図1**）．

今，初期値は十分大きな値（ここでは100）とします．そのノードに到達するたびに，そのノードに到達する前のノードの値にコスト［cost(*n*, *m*)］を足した値をそのノードの値とします．そして，別のルートでそのノードに到達した場合でかつ，ノードの値が小さければ更新します．これをif thenでルールを書くと以下となります．

if【ノードの値：g(*i*)】より【そのノードに到達する前のノードの値：g(*now*)】に【コスト：cost(*now*, *i*)】を足した値が小さければ（①）
then g(*i*)をg(*now*) + cost(*now*, *i*)に更新する（②）

ここで*now*は現在対象としているノードの番号で，*i*はそのノードにつながっているノードの番号の内の

プログラムは本誌ダウンロード・ページから入手できます．
https://www.cqpub.co.jp/interface/download/contents2024.htm