巡回セールスマン問題を解きたい！

BV21013 堀毛晴輝

2023 年 5 月 16 日

1 はじめに

巡回セールスマン問題という問題をご存知でしょうか．比較的有名な問題なので，聞いたことがある人も多いのではないでしょうか．

今回は，この問題について厳密解を求めるアルゴリズムと，近似解を効率よく求める手法について紹介します．

2 巡回セールスマン問題とは？

与えられた都市の集合に対して，すべての都市を一度だけ訪れて出発点に戻る最短経路を求める問題を**巡回セールスマン問題**といいます．

N 個の都市があるとき，すべての都市を巡回して戻ってくるルートは $N!$ 通り存在します．すべての経路を試して最短経路を求めようとすると，

$N = 10$ のとき:	3628800	通り
$N = 20$ のとき:	2432902008176640000	通り
$N = 100$ のとき:	$9.3326\ldots \times 10^{157}$	通り

について経路の長さを求めることになり，N が大きいと愚直解法はとても困難です．

3 アルゴリズム的なアプローチ

先述した “すべての経路を愚直に試し，最短経路を求める” という手法だと，$\Omega(N!)$ かかりますが，ヘルドカープのアルゴリズムを用いると，$O(N^2 2^N)$ で解くことができます．

大体 10^8 回くらいの処理が 1 秒に行えると仮定すると，$N = 20$ のとき愚直解法では 約 15429 年くらい[*1]かかりますが，ヘルドカープのアルゴリズムを用いると**約 4 秒**で求めることができるようになります．

3.1 ヘルドカープのアルゴリズム

N 個の都市がある巡回セールスマン問題，つまり，頂点集合 $V = \{1, 2, \ldots, N\}$ に対して 1 からスタートし，V の頂点をすべて訪れて 1 に戻る最短経路を求める問題を考えます．

以下，頂点 u から頂点 v への距離を $d_{u,v}$ と表します．

$$f(S, v) := 1 \text{ からスタートし } S \text{ の各頂点をすべて訪れ，} \\ v \text{ にいる最短経路の長さ}$$

という関数を考えます．ここで，S は V の部分集合であり，$v \in S$ とします．この関数の値が求められたら，解きたい問題の解は，$\min_{v \in V} \{f(V, v) + d_{v,1}\}$ です．

この関数 $f(S, v)$ を求めたいです．まず，$f(\{1\}, 1) = 0$ です (1 からスタートして今 1 にいるので，距離は 0 です)．

S が $\{1\}$ でないときの $f(S, v)$ の値を考えます．

$S \backslash \{v\}$ について，$s \in S \backslash \{v\}$ とすると，$f(S \backslash \{v\}, s)$ は，“$S \backslash \{v\}$ の各頂点をすべて訪れて s にいるときの最短経路の長さ” でした．

$f(S \backslash \{v\}, s)$ から v に移動することを考えると，その値 $f(S \backslash \{v\}, s) + d_{s,v}$ は “S の各頂点をすべて訪れて v にいる最短経路の長さの**候補**” になります．求める値は候補のうちでもっとも小さいものになるので，

$$f(S, v) = \min_{s \in S \backslash \{v\}} \{f(S \backslash \{v\}, s) + d_{s,v}\}$$

で求めることができます．

求めたい値は $\min_{v \in V} \{f(V, v) + d_{v,1}\}$ だったので，先ほどの式を用いて求めればよいです．計算量は $O(N^2 2^N)$ になります．

4 ヒューリスティック的なアプローチ

先ほどのヘルドカープのアルゴリズムでは，$N \leq 20$ 程度までなら実用的な時間で解くことができますが，それを超えると指数的に実行時間が長くなるため，解くことが困難です．

近似解を求めることを考えます．ここでいう近似解とは，「最適解には達しないが，比較的良い解」のことです．

ある問題に対して，現在の解を少し変化させて良くなったら採用し，これを繰り返すことで答えをどんどん良くしていく手法を**山登り法**といいます．

巡回セールスマン問題に対しても，山登り法を用いることで近似解を求めることができます．具体的に，以下のような手順で近似解を求めます．

1. ランダムな順列を生成する (初期解)
2. 以下を繰り返す
   (a) 現在の解を少し変化させる (近傍解)
   (b) 現在の解より近傍解のほうが良ければ，近傍解を現在の解とする

ここでは，近傍解を求める方法を「ランダムに 2 つの都市を選び，訪れる順番を入れ替える」として考えます．

図 1 ランダム 2 点間スワップ

ランダムな初期解から時間が経つにつれて，解が良くなっていくことがわかります．これが山登り法です．

山登り法は，最適解に到達する保障はありませんが，近傍解を効率的に生成できる場合，比較的シンプルに近似解を求めることができます．しかし，局所最適解に陥りやすかったり，初期解や近傍解の生成方法によっては良い解にたどり着けない場合もあります．

4.1 2-opt

先ほどは近傍解の生成方法を「ランダムに 2 つの都市を選び，訪れる順番を入れ替える」としましたが，このほかにも様々な近傍解の作り方が考えられます．

ここでは，近傍解の生成方法の 1 つである **2-opt** について紹介します．

2-opt とは，現在の解の中で 2 つの都市を選び，その間の順番を逆にすることで近傍解を作ります．この操作により，交差しているルートを解消でき，より効率的に近似解を求めることができます．

図 2 2-opt

参考文献

[1] ビット DP(bit DP) の考え方 集合に対する動的計画法 ，https://algo-logic.info/bit-dp/, 2023 年 5 月 16 日閲覧．

[2] Qiita 2-opt の実装，https://qiita.com/hotpepsi/items/424f9491e7baaa63b6ce, 2023 年 5 月 16 日閲覧．

[*1] $N!$ 通りのルートに対して，N 回の計算をして長さを求める，つまり $\Theta(N \cdot N!)$ のとき