プログラミング演習 第 10 回演習レポート

2064229 福場脩真

# 指定された課題番号と課題名

[演習課題 4. 経路探索の解を求めるプログラムの改良](javascript:void(0))

# (2) フローチャートあるいは自然言語によるアルゴリズムの記述

Route\_list = 道路綱のリスト

Node\_num = 道路綱リストの長さ

Unsearched\_nodes = node\_numのレンジでとったリスト

Distance = [math.inf] とnode\_numの積

Previous\_nodes = [-1]とnode\_numの積

Distance[0] = 0

関数 get\_target\_min\_index(min\_index, distance, unsearched\_nodes):

    start = 0

    while True:

        index = distance.index(min\_index, start)

        found = index in unsearched\_nodes

        if foundに値が入っている:

            return index

        else:

            start = index + 1

while unsearched\_nodesの長さが０でないとき:

    posible\_min\_distance = math.inf

    for unsearched\_nodes の中身をnode\_indexとしてループ処理:

        if posible\_min\_distance が distance[node\_index]　より大きいとき:

            posible\_min\_distance に distance[node\_index]を代入

    target\_min\_index に 関数get\_target\_min\_index(posible\_min\_distance, distance, unsearched\_nodes)の返り値を代入

    unsearched\_nodesからtarget\_min\_indexを取り除く

    target\_edge に route\_list[target\_min\_index] を代入

    for target\_edgeを要素番号index, 要素をroute\_disとしてループ処理:

        if route\_dis が0でないとき:

            if distance[index] がdistance[ target\_min\_index] と route\_disの和より大きい:

                distance[index] に distance[ target\_min\_index] + route\_disを代入

                previous\_nodes[index] に target\_min\_indexを代入

print("-----経路-----")

previous\_node に node\_num – 1を代入

while previous\_node が-1でないとき:

    if previous\_node が0でないとき:

        previous\_node + 1を文字列型として、 -でつないで出力

    else:

        (previous\_node + 1)を文字列型として出力

    previous\_node　に previous\_nodes[previous\_node]を代入

# アルゴリズムが「正しいこと」である説明あるいは証明

プログラムを動作させたところ、プログラムはエラーを起こさず正しく停止し、最短経路を正しく出力した。よってこのプログラムは正しい。

# ソース・プログラムの説明

メインの処理では、未探索ノードがなくなるまでwhile文を用いてプログラムを動かすようになっている。まず未探索ノードのうち、distanceが最小のものを選択する。初期値は1として、未探索んーどのループを回した。初期値よりより値が小さければ更新し、最小のノードの要素番号を関数get\_target\_min\_indexで取得する。この後探索を開始するので、未探索リストunsearched\_nodesから最小ノードを削除する。

Target\_listはターゲットになるノードからのびるエッジのリストとして定義した。このリストをループさせ、各要素が0でないかを判定する。0で無いものは経路綱でつながっている。0でなければ、過去に設定されたdistanceよりも小さい場合はdistanceを更新する。ひとつ前に到達するノードのリストも更新しておく。

こうして得られた最短経路を出力する。

# (5) 考察

今回の実装では、再帰関数を用いずプログラムしたので、バックトラックを用いたものへ改良する余地があると考えられる。またプログラム自体も冗長になったため、処理は早かったが再帰を用いてシンプルにまとめられるだろう。