5801

ヒープ機能付ダイクストラ法による最短経路算出

Calculation of the shortest path by Dijkstra's algorithm with heap function

CS23　中村涼平
指導教員　小出由起夫

1. 研究背景

グラフ理論において,ネットワーク上の全ノードへの最短経路を算出するアルゴリズム「ダイクストラ法」が存在する.ダイクストラ法は実道路ネットワークのデータなど大規模なネットワークで使用すると膨大な計算時間を必要とするため実用的ではない.したがって高速化が必要不可欠である.

2. 研究目的

本研究ではバイナリヒープを使用してダイクストラ法を高速化し,大規模な実道路ネットワーク上での最短経路の算出を行う.これによりグラフ理論における最短経路問題について知見を広げる.

3. 研究方法

ダイクストラ法は任意の始点に繋がるノードから順に最短経路を確定していく. 次に進むべきノードには,現在地に繋がるノードの中から,始点からの距離が最も短いノードが選ばれる.通常のダイクストラ法ではループを使用してこのノードを算出しており,非常に効率が悪い.したがって,ダイクストラ法において高速化が必要となるのは, 各ノードへの最短距離の更新・次に進むノードの抜き出しである.

本研究ではヒープの中でも特に構造が単純なバイナリヒープを使用してこれらの問題を解消する.バイナリヒープは親ノードよりも子ノードのほうが大きいか等しいという特徴をもつ.ダイクストラ法の処理の中で仮の最短距離が更新される度にその値をヒープの末尾に挿入し,上方移動することでヒープ性を保つことができる.このときルートノードは最短距離の中で最も小さい値をもち,これが次に進むべきノードとなる.

以上の機能を実装したダイクストラ法・実装しないダイクストラ法でそれぞれ最短経路を算出し,計算に要した時間を求める.これにより各プログラムの性能にどれほどの違いが現れるかを検証する.

表 1 にプログラムの作成・実行に使用した環境を示す.

表 1. プログラム開発環境

項目	内容
CPU	Intel Core 2 DuoE7400
メモリ	4.00GB
OS	Windows 7 Professional
開発言語	Visual C++

4. 結　果

最短経路算出には 4 種類の実道路ネットワークのデータを使用する.表 2 にこれらのデータで最短経路を算出した際の必要計算時間を示す.

表 2.各データにおける必要計算時間

ノード数	エッジ数	ヒープなし	ヒープあり
218	676	0.012 秒	0.011 秒
2973	6412	0.140 秒	0.076 秒
9901	29702	1.647 秒	0.463 秒
25092	74220	9.797 秒	1.408 秒

表 2 を見ると,ノード数が少ないうちはヒープなしのダイクストラ法でも大した計算時間を必要としていないことが分かる.しかし,ネットワークの規模の肥大化に伴い,必要とされる計算時間は格段に増えている.そのため,更に規模の大きなネットワークで処理を行った場合,ますます計算時間が必要になると推測できる.

一方,ヒープを使用したダイクストラ法はノード数・エッジ数がいくつであっても比較的高速に処理を行えていることが分かる.このことより, 高速化したダイクストラ法の有意性が確認できる.

5. まとめ

本研究ではバイナリヒープを使用したダイクストラ法,使用しないダイクストラ法プログラムを作成し,それぞれで実道路ネットワークにおける最短経路の算出を行った.このとき,ヒープを使用したダイクストラ法の計算時間が短くなることも確認した.

また,本研究ではノード間のコストに単純な距離を使用した.しかし,実際の道路では道の険しさ,かかる費用なども考慮して使用する道を選択する必要がある.そのため,今後この研究を続ける場合,これらについても考慮したプログラミングを行うと良い.

文　献

[1] 浅野孝夫,“基本データ構造”,情報の構造,上, pp.45-51, (Mar.1994)
[2] 浅野孝夫,“ネットワークアルゴリズム”,情報の構造,下, pp.240-246, (Mar.1994)
[3] 秋葉拓哉, 岩田陽一, 北川宜稔, “単一始点最短路問題 2(ダイクストラ法),” プログラミングコンテストチャレンジブック [第 2 版], pp.96-97, (Jan.2012)