ダイクストラアルゴリズムによる経路探索

山口 真実　(指導教員：萩田真理子)

1　はじめに

連結グラフ $G=(V,E)$ の各辺 $e \in E$ に重み w が与えられたものを重みつき連結グラフという。この重みつき連結グラフにおいて、頂点 $s \in V$ から頂点 $t \in V$ までの独立な 2 本のパスを見つけたい。

重みの和が少ない 1 本のパスを見つけるためのアルゴリズムとしては、ダイクストラのアルゴリズムが知られている。また、指定した本数の独立パスがとれるための必要十分条件を与える定理として、Mengere の定理が知られている。

本発表では、この重みつき連結グラフにおいて、頂点 $s \in V$ から頂点 $t \in V$ までの 2 本のパスの重みが最小となる独立パスを見つけることを目標とする。

2　ダイクストラアルゴリズム

重みつきの連結グラフ $G=(V,E)$ において、頂点 s からの重み最小のパスを与えるアルゴリズムとして、ダイクストラアルゴリズムが有名である。

[ダイクストラアルゴリズム]
重みつきグラフ G で、全ての頂点 $v \in V$ に対し、s から v へのパスがいつでも最短距離となる全域木を構成するアルゴリズム。

入力：グラフ G、始点 s

1. 始点 s から、s の隣接点までの距離を求める。始点 s の距離は 0 とする。この距離は以後変更がないので、距離が確定した点とする。
2. 最短距離がまだ確定していない点のうち、一番距離が小さい点 k を見つける。点 k の重みつき距離を確定。
3. 仮最短距離を更新する。
4. 全ての点の重みつき距離が確定するまで、step2, 3 を繰り返す。

出力：始点 s から各頂点までの重みつき距離

このアルゴリズムは、次に示すダイクストラの定理によって重み最小のパスを得られることが証明されている。

[ダイクストラの定理]
$G=(V,E)$ を連結グラフとし、$w : E \to \mathbb{R}^+$ を重み関数とする。このときダイクストラアルゴリズムは、全ての頂点 $v \in V$ に対して、「s から v への重みつきの距離：$d(s,v)=$ パス v の重みつきの長さ：$l(v)$ である G において、s から v への一意的なパスが、最短の $s, v-path$ である」という特徴を持つ全域木を与える。

このダイクストラアルゴリズムを用いて、頂点 s から頂点 t への 2 本の独立なパスをみつける。独立なパスというのは、同じ辺や頂点を用いていないパスのことである。

3　Mengere の定理

Mengere の定理とは、連結な無向グラフにおいて、k 本の独立パスが存在する条件を与える定理である。

[Mengere の定理 (無向グラフ)]
G を連結な無向グラフとし、s、t をグラフ G の隣接でない 2 頂点とする。グラフ G において、「独立な $s-t$ パスの本数の最大値 k」は、頂点 s,t の「頂点カットの点の数の最小値」に等しい。

この定理から、1 点だけでグラフを非連結にしてしまう点 (切断点) がなければ、グラフ G 上で 2 本の独立パスが存在することがわかる。

本発表で扱うグラフは、**切断点が存在しないグラフ**とする。

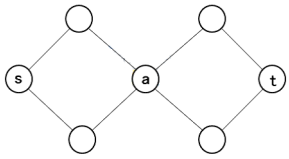

図 1: 2 本の独立パスが存在しない例

図 1 では、頂点 a が切断点となっている。このため、頂点 s から頂点 t への 2 本の独立パスは存在しないことがわかる。

4　片方優先アルゴリズム

2 本とるパスのうち、片方のパスとして、最短距離のパスを用いるアルゴリズム。もう一方のパスは、最短距離のパスを取り除いたグラフから見つける。

入力：グラフ G、始点 s、終点 t

1. グラフ G において、頂点 s をスタート地点とし、ダイクストラアルゴリズムを実行する。
2. 1 つ目のパスは、頂点 t までのパス P とする。
3. $G' = G \setminus P$ とする。そして、頂点 s をスタート地点とし、グラフ G' において、ダイクストラアルゴリズムを実行する。
4. 2 つ目のパスは、t までのパス P' とする。

出力：始点 s から終点 t までの 2 本の独立パス

本アルゴリズムにより 2 本の独立パスを探すと、1 つ目のパスは最短経路になる。しかし、2 つ目のパスは長い経路になることが多い。そして、1 つ目のパスを見つける時に多くの辺を用いると、残りの辺だけでは、頂点 t までの経路がなくなってしまい、2 つ目のパスが見つからなくなってしまうこともある。

このことを踏まえ、1 本のパスに用いる辺の本数を出来るだけ少なくするようなアルゴリズムを次に示す。

5　距離優先アルゴリズム

与えられた重みつき連結グラフの重みを全て 1 として考え、探索するアルゴリズム。

入力：グラフ G、始点 s、終点 t

1. 与えられた重みつきグラフ G の重みを全て 1 としたグラフ H を考える。
2. グラフ H 上で、頂点 s をスタート地点とし、ダイクストラアルゴリズムを実行する。
3. 1 つ目のパスは、頂点 t までのパス P とする。
4. $H' = H \setminus P$ とする。そして、頂点 s をスタート地点とし、グラフ H' において、ダイクストラアルゴリズムを実行する。
5. 2 つ目のパスは、t までのパス P' とする。

出力：始点 s から終点 t までの 2 本の独立パス

本アルゴリズムでは、頂点 s から頂点 t にへの辺のうち、最小の数の辺だけを使っている。このため、1 つ目のパスをみつけるときにたくさんの辺を使ってしまうということはないので、2 本の独立パスをみつけやすい。

6　片方優先アルゴリズムと距離優先アルゴリズムの比較

改めて、片方優先アルゴリズムと距離優先アルゴリズムの比較を行う。そのための例として次の図を考える。

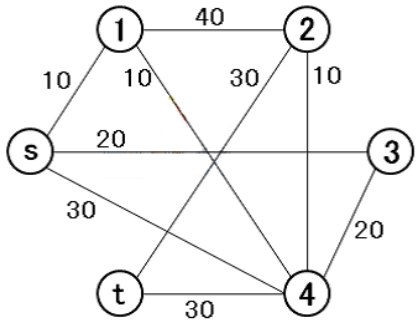

図 2: 重みつき連結グラフの例

1. 片方優先アルゴリズムによる結果
   1 つ目のパス：$s \to 1 \to 4 \to t$
   2 つ目のパス：not found.
2. 距離優先アルゴリズムによる結果
   1 つ目のパス：$s \to 4 \to t$
   2 つ目のパス：$s \to 1 \to 2 \to t$

図 2 に示したグラフでは、片方優先アルゴリズムでは 2 本の独立パスを見つけることが出来なかったが、距離優先アルゴリズムでは 2 本の独立パスを見つけることが出来た。これは、先にも述べたが、片方優先アルゴリズムでは、一方のパスを取るときに、重みを最小にするため、多くの辺を使ってしまうことが原因となっている。

片方優先アルゴリズムよりも、距離優先アルゴリズムのほうが、2 本の独立パスを取れる可能性が高いことは自明である。しかし、このアルゴリズムを用いた場合でも、2 本の独立パスを取れるグラフ、つまり切断点が存在しないグラフの場合でも 2 本の独立パスが取れないことがある。そして、距離優先アルゴリズムでは、辺の重みについて全く触れていないので、距離が無駄に大きくなってしまうこともあり得る。

以上のことを踏まえ、次のアルゴリズムを考えた。

7　距離を優先とし、2 本のパスを同時に確保するアルゴリズム

ここでは、パスに使用する辺の本数が少ないほど、パスに用いた辺の重みが小さくなるようなグラフを考える。このようなグラフを考えることによって、距離を優先にグラフ G を検索していっても、より重みの少ない辺を通るために、探索してきた経路を戻って探索し直す必要がなくなる。

入力：グラフ G、始点 s、終点 t

1. s からの距離別に頂点を分ける。
2. t からの距離別に頂点を分ける。
3. s と t それぞれからの距離の合計が、s から t への距離と等しいもの以外の点は考えないことにする。(使わない点を削除)
4. s からの距離が 1 のものから順に、距離が 1 番小さいもの 2 つを探す。t 点までの距離を d とする。
5. step2 を距離 $d-1$ まで繰り返す。

出力：始点 s から終点 t までの 2 本の独立パス

本アルゴリズムと、先に挙げた 2 つのアルゴリズムとの最大の違いは、頂点 t からの距離も考えていることである。頂点 t からの距離を考えることによって、途中までたどったパスが行き止まりになってしまうことを防ぐ。そして、距離のみを考えているのではなく、重みつきでの距離を考えているため、距離優先アルゴリズムで得られるパスよりも重みの少ないパスを見つけることが出来る。

8　まとめと今後の課題

グラフ G における頂点 s から t への 2 本の独立パスを見つけるためのアルゴリズムを考案した。距離を優先とし、2 本のパスを同時に確保するアルゴリズムが適応できる条件を広め、より多くの独立パスを見つけられるアルゴリズムを考案したい。そして、重みの和が少ないの 2 本の独立パスを得るための条件と、本アルゴリズムで得られる 2 本のパスの重みの最小値からの誤差の最大値を考察したい。

参考文献

[1] Martin Aigner : Discrete Mathematics

[2] J.A.Bondy・U.S.R.Murty : Graph Theory