青 山 学 院 大 学 理 工 学 部
経 営 シ ス テ ム 工 学 科

卒 業 論 文

多目的最短経路問題における動的計画法に基づいた拡張ベルマンフォード法の提案

２０１８年度

中野　壱帥
15715051

指導教員　宋 少秋　印

目 次

第 1 章　はじめに **3**

1.1　研究背景 3

1.2　研究目的 4

1.3　章構成 4

第 2 章　諸定義 **5**

2.1　多目的最適化問題 5

2.2　パレート最適解 5

2.3　最短経路問題 6

2.4　多目的最短経路問題 10

2.5　関連研究 11

第 3 章　従来研究 **12**

3.1　拡張ダイクストラ法 12

3.2　完全多項式時間近似スキームによる解法 13

3.3　ラベル付けアルゴリズムによる解法 15

第 4 章　解法の提案と実験的評価 **18**

4.1　問題の分析 18

4.2　非負数の問題に対する解法の提案 18

4.2.1　提案解法 1 18

4.2.2　提案解法 2 20

4.3　負の値の考慮した問題に対する解法の提案 22

4.4　実装における工夫 22

4.5　提案解法の評価と分析 22

第 5 章　結論 **23**

5.1 研究成果 23
5.2 今後の課題 23

第1章　はじめに

1.1　研究背景

現代には道路ネットワークや通信ネットワークなど様々なネットワークが存在する．これらのネットワークには無数の経路や組み合わせが存在するため最適化した解を求めたい．しかし，これらのネットワークに対する最適化を行う場合，複数の目的関数を考慮することが必要である．例えば，道路ネットワークでは目的地までの時間とコストを最小化する必要がある．このように複数の目的関数値を最大化（最小化）する解を求める問題は多目的最適化問題と呼ばれている．多目的最適化問題の中でも，最短経路を求める問題は多目的最短経路問題と呼ばれている．

多目的最適化問題においては，それぞれの目的関数がトレードオフの関係にある場合が存在し，全ての目的関数値が最大（最小）となる最適解が存在するとは限らない．あらかじめ各目的関数に対する比重を決めて解を求めると１つの解が適切に求めることができる．しかし，探索者の各目的関数に対する比重が決まっていない場合や複数の探索者がいる場合，求まった１つの解がそれぞれの探索者に対して適切な解ではない場合がある．そこで，最適解になり得るパレート最適解の集合を求める．探索者が１つの解を求めるとき，パレート最適解の集合を求めることにより解を選択する意思決定を容易にできる．また，それぞれの探索者が最適解となり得る解の集合から解を選択できるので，それぞれの探索者が最適な解を求めることができる．一般にパレート最適解は膨大な数存在するので効率的に列挙することが必要になる．

単一目的最短経路問題には負の要素を考慮した解法が提案されているが，調査した限り従来の多目的最短経路問題には負の要素を考慮した研究がなされていない．単一目的最短経路問題において負の要素を考慮した場合，負のサイクルが存在し解が求められない場合がある．しかし，多目的最短経路問題において負の要素を考慮した場合，１つの目的関数において負のサイクルが存在しても他の目的関数に対する解を求められる場合がある．そこで，負の要素を含む多目的最短経路問題において負のサイクルが存在しない目的関数に対する効率的な解法を考える．

1.2　研究目的

本研究では，多目的最短経路問題において一般的に膨大な数存在するとされるパレート解を効率的に列挙する．従来研究で提案されている拡張ダイクストラ法等にはインスタンスによって探索順序が適切でなかったり実装するにあたって改善点が多くある．また，従来研究では負の要素を考慮していないため負のサイクル存在時の解の定義がされていない．本研究では，オフライン環境における多目的最短経路問題に対して以下を目的とする．

目的１：多目的最短経路問題に対する効率的解法を行う．

多目的最適化問題におけるパレート解列挙の複雑さを踏まえて，解法の提案を行う．また，計算機を用いて解法の実験的評価を行う．

目的２：負の要素を考慮した解法を行う．

負のサイクルが存在する場合の解を定義し効率的な解法の提案を行う．また，計算機を用いて解法の実験的評価を行う．

1.3　章構成

本論文の章構成は以下である．

- 第２章では，多目的最適化問題と最短経路問題に対する定義を紹介し，多目的最短経路問題の解に対する定義を行う．
- 第３章では，多目的最短経路問題に対する従来研究を紹介する．
- 第４章では，提案解法の紹介と実装における工夫，本研究に対する成果を述べる．
- 第５章では，結論として本研究の成果と今後の課題について述べる．

第2章　諸定義

この章では，多目的最適化問題と最短経路問題に対する定義を紹介する．多目的最適化問題における解となるパレート解の説明をする．単一目的最短経路問題に対する解法を紹介する．また，多目的最短経路問題の解に対する定義を行う．本研究における定式化をする.（負の要素を考慮した場合を含む）

2.1　多目的最適化問題

最適化問題とは与えられたインスタンスに対して実行可能な最適解を求める問題である．最適化問題のインスタンスは特定の集合上で定義された実数値関数または整数値関数 $f : A \to \mathbb{R}$ で定義される．最小化問題の場合，$minf(x)$ となる x を求める．すなわち，$x_0 \in A : \forall x \in A, f(x_0) \leq f(x)$ となる x_0 を求める．最大化問題の場合，$maxf(x)$ となる x を求める．

多目的最適化問題とは複数の目的関数に対する最適化問題である．多目的最適化問題のインスタンスは最適化目的の数が n である場合，n 個の実数値関数または整数値関数 $f : A \to \mathbb{R}$ で定義される．すなわち，$\vec{f} = (f_1, \ldots, f_n)$ と表される．最小化問題の場合，$minf(\vec{x})$ となる x を求める．多目的最適化問題の場合，それぞれの目的関数がトレードオフの関係にある場合が存在し，全ての目的関数値が最大（最小）となる最適解が存在するとは限らない.（例：$f_1(x_0) < f_2(x_0) \wedge f_1(x_1) > f_2(x_1)$) 一般的に，多目的最適化問題はパレート最適解の集合を求める．パレート最適解は複数の目的関数をそのまま考慮された解なので，求めたい選好解を見つけることや挙動変数の関係を知ることが可能になる．

2.2　パレート最適解

パレート最適解は，多目的最短経路問題のにおける解の支配関係により定義される．解 x, y が以下の条件を満たすとき，x は y を支配する．

- $\forall i \in \{1, \ldots, k\}, f_i(x) \leq f_i(y)$

- $\exists i \in \{1, \ldots, k\}, f_i(x) < f_i(y)$

- k：最適化目的の数，f：目的関数

パレート最適解とは取りうる値の範囲を全て考慮した上で支配されない解である．

図 1 に，目的関数の数が 2 の場合のパレート最適解の例を示す．図中の黒丸がパレート最適解を示している．

2.3 最短経路問題

有向もしくは無向グラフをグラフ $G = (V, E)$ と表す．グラフは頂点集合 $V = \{v_1, \ldots, v_n\}$，辺集合 $E = \{e_1, \ldots, e_m\}$ から成る．重み付きグラフの場合，全ての辺 $e \in E$ は重み e_w を持つ．無向グラフにおいて，両端点を s, t とする辺 $e \in E$ を $s, t \in e$ と表す．有向グラフにおいて，始点を s，終点を t とする辺 $e \in E$ を $e = (s, t)$ と表す．s から t までの距離を $d_{s,t}$ と表す．最短経路問題とは重み付きグラフ $G = (V, E)$ の与えられた 2 つのノード s, t 間を結ぶ経路の中で，重みが最小の経路 ($min\ d_{s,t}$ となる経路) を求める最適化問題である．

最短経路問題の種類

- 2 頂点対最短経路問題：特定の 2 つのノード間の最短経路問題.

  入力： 重み付きグラフ，始点 s，終点 t

  出力： s から t への最短経路

- 単一始点最短経路問題:特定の 1 つのノードから他の全ノードとの間の最短経路問題.

  入力： 重み付きグラフ，始点 s

  出力： s から全頂点への最短経路

- 全点対最短経路問題：グラフ内のあらゆる 2 ノードの組み合わせについての最短経路問題.

  入力： 重み付きグラフ

  出力： 全頂点間の最短経路

最短経路問題の主な解法

- 幅優先探索

  始点から近い順に探索する．重みがない（すべての重みが 1 である）最短経路問題に使われる．通った辺の本数に応じて重みが決まるため探索によって発見した頂点は最短経路が決定する．計算時間は $O(E)$ である．以下に無向グラフにおける単一始点最短経路問題の解法を示す．

  入力： グラフ $G = (V, E)$，始点 $s \in V$，$Q \leftarrow \emptyset$

  出力： s から全ての頂点への経路

  Step 1. $Q \leftarrow s$

  Step 2. $Q = \{\emptyset\}$ になるまで以下の操作を行う．

  Step 2-1. Q の先頭にあるノード v を取り出す．

  Step 2-2. $u = \{u \in V \mid v, u \in e \wedge e \in E\}$ が未探索のとき，$Q \leftarrow u$.

  Step 3. 経路を出力する．

  2 頂点対最短経路問題の場合，終点が見つかった時点で探索を終了し，始点から終点への経路を出力する．入力が有向グラフの場合：

  Step 2-2. $u = \{u \in V \mid e = (v, u) \wedge e \in E\}$ が未探索のとき，$Q \leftarrow u$.

- ダイクストラ法

  全ての重みが非負数であるグラフについての有名なアルゴリズムであるダイクストラ法について説明をする．ダイクストラ法はエドガー・ダイクストラ [?] によって開発された．ダイクストラ法は全ての重みが非負数であるグラフにおいて使われる．探索手順はすでに探索済みのノードの中で重みが最小のノードを求め，更新対象として探索していく．全ての重みが非負数の場合，探索したノードの中で重みが最小のものはその後の探索で更新されることはないので重みが決定する．ダイクストラ法の計算時間は $O(V^2)$ である．以下に無向グラフにおける単一始点最短経路問題の解法を示す．

  入力： グラフ $G = (V, E)$，始点 $s \in V$，E の各辺の長さ，$Q \leftarrow \emptyset$

  出力： s から全ての頂点への経路

  Step 1. $s_w = 0$ とし，$v \in V/\{s\}$ に対して $v_w = \infty$ とする．

  Step 2. $Q \leftarrow s$

Step 3. $Q = \{\emptyset\}$ になるまで以下の操作を行う.

Step 3-1. $v = \{v \in Q \mid v_w \leq v'_w \wedge v' \in Q\}$ を取り出す.

Step 3-2. $e = \{e \in E \mid v \in e\}$ について以下の操作を行う.

Step 3-2-1. $u = \{u \in V \mid u \in e\}$ となる u に対して，$u_w > v_w + e_w$ を満たすとき以下の操作を行う.

Step 3-2-1-1. $u_w = v_w + e_w$.

Step 3-2-1-2. $u \notin Q$ のとき，$Q \leftarrow u$.

Step 4. 経路を出力する.

2 頂点対最短経路問題の場合，終点が更新対象となった時点で探索を終了し，始点から終点への経路を出力する．入力が有向グラフの場合:

Step 3-2. $u = \{u \in V \mid (v, u) = e \wedge e \in E\}$ となる u に対して，$u_w > v_w + e_w$ を満たすとき以下の操作を行う.

Step 3-2-1. $u_w = v_w + e_w$.

Step 3-2-2. $u \notin Q$ のとき，$Q \leftarrow u$.

- ベルマンフォード法

  グラフ内の全ての重みが非負数であるかどうかに関わらず使用できる有名なアルゴリズムであるベルマンフォード法について説明をする．ベルマンフォード法はリチャード・E・ベルマンと Lester Ford, Jr. [?] によって開発された．全ての重みが実数であるグラフにおいて使われる.（負の値を含んでいても使用できる）ベルマンフォード法の探索順序は頂点数を $|V|$ とした時，全辺を緩めることを単に $|V| - 1$ 回繰り返す．辺 e を緩めるとは，e に接続しているノードの重みを v_w, u_w とし，e の重みを e_w としたとき，u_v が $v_w + e_w$ によって更新されることである．$|V| - 1$ 回の探索終了時にもう一度全辺を緩め，更新が行われた場合は負の閉路が存在するため負の閉路の存在を報告する．負の閉路が存在しない場合は path の長さの最大が全頂点を通る $|V| - 1$ となる．負の閉路が存在する場合は無限に更新が行われるため，$|V|$ 回目の探索でも更新が行われる．よって，$|V|$ 回目の探索により負の閉路の存在有無を確かめられる．ベルマンフォード法の計算時間は $O(EV)$ である．以下に無向グラフにおける単一始点最短経路問題の解法を示す.

  入力： グラフ $G = (V, E)$，始点 $s \in V$，E の各辺の長さ

出力： s から全ての頂点への経路，負の閉路の存在有無

Step 1. $s_w = 0$ とし，$v \in V/\{s\}$ に対して $v_w = \infty$ とする．

Step 2. $|V| - 1$ 回以下の操作を行う．

Step 2-1. $e = e \in E \mid v, u \in e$ となる e に対して，$u_w > v_w + e_w$ を満たすとき以下の操作を行う．

Step 2-1-1. $u_w = v_w + e_w$.

Step 3. Step 2-1 を行いノードの重みが更新された場合，負の閉路の存在を報告する．

Step 4. 経路を出力する．

2 頂点対最短経路問題の場合，単一始点最短経路問題を解き，始点から終点への経路を出力する．入力が有向グラフの場合:

Step 2-1. $e = e \in E \mid (v, u) = e$ となる e に対して，$u_w > v_w + e_w$ を満たすとき以下の操作を行う．

Step 2-1-1. $u_w = v_w + e_w$.

- ワーシャルフロイド法

グラフ内の全ての重みが非負数であるかどうかに関わらないかつ全点対最短経路をもとめる際に使用できる有名なアルゴリズムであるワーシャルフロイド法について説明をする．ワーシャルフロイド法はスティーブン・ワーシャルとロバート・フロイド [?] によって開発された．ワーシャルフロイド法は重み付き有向グラフにおいて全点対最短経路を多項式時間で解くアルゴリズムである．3 つの頂点 a, b, c を選んで、a → b → c という道が a → c という道より短ければ a → c の距離を更新するという操作を全ての頂点の組み合わせで繰り返して最短距離を確定させていく．a, b, c はそれぞれ $|V|$ の選択が出来るので，a → b → c という道が a → c という道より短ければ a → c の距離を更新するという操作は $|V|^3$ 回行われる．$|V|^3$ 回の操作後，更新できる解が存在する場合は負の閉路が存在する．ワーシャルフロイド方は負の値にも対応でき，計算時間は $O(V^3)$ である．以下に全点対最短経路問題の解法を示す．

入力： グラフ $G = (V, E)$，始点 $s \in V$，E の各辺の長さ

出力： 全頂点対の経路

Step 1. $s_w = 0$ とし，$v \in V/\{s\}$ に対して $v_w = \infty$ とする．

Step 2. 各 $1 < k < |V|$ に対して以下の操作を行う．

Step 2-1. 各 $1 < i < |V|$ に対して以下の操作を行う．

Step 2-1-1. 各 $1 < j < |V|$ に対して以下の操作を行う．

Step 2-1-1-1. $d_{i,j} > d_{i,k}+dk, j$ を満たすとき，$d_{i,j} = d_{i,k}+dk, j$

Step 3. 経路を出力する．

2 頂点間の距離がマイナスの場合，負の閉路が存在する．

2.4 多目的最短経路問題

多目的最短経路問題とは，最短経路問題の最適化目的の数を複数にすることによって，多目的最適化問題に拡張した問題である．つまり，目的値が複数の重み付きグラフにおいて与えられた 2 つのノード間を結ぶ経路の中で，パレート最適解となる経路を求める最適化問題である．有向もしくは無向グラフをグラフ $G = (V, E)$ と表す．グラフは頂点集合 $V = \{v_1, \ldots, v_n\}$，辺集合 $E = \{e_1, \ldots, e_m\}$ から成る．最適化目的の数を k とするとき，全ての辺 $e \in E$ は重み $\vec{e_w} = \{e_{w1}, \ldots, e_{wk}\}$ を持つ．無向グラフにおいて，両端点を s, t とする辺 $e \in E$ を $s, t \in e$ と表す．有向グラフにおいて，始点を s，終点を t とする辺 $e \in E$ を $e = (s, t)$ と表す．s から t までの距離を $\vec{d_{s,t}} = \{d_{s,t1}, \ldots, d_{s,tk}\}$ と表す．また，本研究では全ての値が同じとなる経路は 1 つのみ求める．

多目的最短経路問題の種類

- 多目的 2 頂点対最短経路問題：特定の 2 つのノード間の最短経路問題.

入力： 重み付きグラフ，始点 s，終点 t，最適化目的の数 k

出力： s から t へのパレート最適解となる経路の集合

- 多目的単一始点最短経路問題：特定の 1 つのノードから他の全ノードとの間の最短経路問題.

入力： 重み付きグラフ，始点 s，最適化目的の数 k

出力： s から全頂点へのパレート最適解となる経路の集合

- 多目的全点対最短経路問題：グラフ内のあらゆる 2 ノードの組み合わせについての最短経路問題.

入力： 重み付きグラフ，最適化目的の数 k

出力： 全頂点間のパレート最適解となる経路の集合

負の値を含む場合

単目的最短経路問題では負の閉路が存在する場合，負の閉路の存在を報告し最短経路は求めなかった．これは負の閉路を何度も通過することによって重みを更新し続けるためである．しかし，多目的最短経路問題の場合，目的関数が複数存在するので 1 つの目的関数において負の閉路が存在する場合でも，他の目的関数による最適化をすることで解を求めることができる．よって，本研究では負の閉路が存在する目的関数を考慮しない解を求める．

2.5 関連研究

第3章　従来研究

この章では多目的最短経路問題に対する従来研究の成果を紹介する．

3.1　拡張ダイクストラ法

拡張ダイクストラ法とは頂点 v を指定し，v に隣接する頂点 u に対する解を更新していく方法である．また，いくつかの工夫によって探索空間を削除することが可能である．拡張ダイクストラ法は全ての値が非負数の場合のみ使用できる．以下に無向グラフに対するアルゴリズムを記載する．

記号

k：最適化目的の数

$v \in V$ に対して

W_v：ノード v に隣接するノード集合

$v \in V$，$j = 1, \ldots, k$ に対して

l_{jv}：始点からノード v に到達したときに生じる第 j 番目の目的関数における総コスト

$e \in E$，$j = 1, \ldots, k$ に対して

e_{jw}：辺 e の第 j 番目の目的関数におけるコスト

アルゴリズム

入力：グラフ $G = (V, E)$，始点 $s \in V$，最適化目的の数 k，各辺の重みを返す関数 $w : E \to \mathbb{R}^k$

出力：s から全ての頂点への最短経路となるパレート解の集合

Step 1. $L_v \leftarrow (s, 0, \ldots, 0)$，$W_v \leftarrow \emptyset$，$v \leftarrow s$

Step 2. W_v を求める．

Step 3. W_v 内のノード全てに対して以下の操作を行う．

Step 3-1. $\omega \in W_v$ を選択する．

Step 3-2. $(v, l_{1v}, \ldots, l_{kv}) \in L_v$ を選択する．

Step 3-3. 始点が v，終点が ω である辺 e の重みベクトルを $\vec{e} = (e_{1w}, \ldots, e_{kw})$ とし，$(\omega, l^*_{1\omega}, \ldots, l^*_{k\omega}) \leftarrow (\omega, l_{1v} + e_{1w}, \ldots, l_{kv} + e_{kw})$ とする．

Step 3-4. L_v と $(\omega, l^*_{1\omega}, \ldots, l^*_{k\omega})$ に対してパレート最適解の判定を行う．

Step 3-5. 全ての頂点 $v \in V$ が探索されていないとき，$v' \in W_v$ を選択し，$v \leftarrow v'$ として Step 2 に戻る．

Step 4. 全てのパレート解を出力

パレート最適解の判定

Step 1. L_v と $(\omega, l^*_{1\omega}, \ldots, l^*_{k\omega})$ を受け取る．

Step 2. L_v の全てのラベルに対して以下の操作を行う．

Step 2-1. $(\omega, l'_{1\omega}, \ldots, l'_{k\omega}) \in L_v$ を選択する．

Step 2-2. 以下の条件を満たすとき，L_v を返す．

- $\forall i \in \{1, \ldots, k\}, l'_{i\omega} \leq l^*_{i\omega}$

Step 2-3. 以下の条件を満たすとき，$L_v \leftarrow L_v \setminus \{(\omega, l'_{1\omega}, \ldots, l'_{k\omega})\}$．

- $\forall i \in \{1, \ldots, k\}, l^*_{i\omega} \leq l'_{i\omega}$
- $\exists i \in \{1, \ldots, k\}, l^*_{i\omega} < l'_{i\omega}$

Step 3. $L_v \leftarrow L_v \cup \{(\omega, l^*_{1\omega}, \ldots, l^*_{k\omega})\}$ を返す．

3.2 完全多項式時間近似スキームによる解法

完全多項式時間近似スキームとは，入力サイズが n，精度が $1/\epsilon(\epsilon > 0)$ となる多項式時間アルゴリズムである．また，任意の整数 $\epsilon > 0$ に対して $\alpha = 1 + \epsilon$ とできる入力サイズの多項式時間アルゴリズムを多項式時間近似スキームという．今回のアルゴリズムはラベルを 2 つ用意し，それぞれ以下とする．

- パレート解となり得るかつ更新対象としない解のラベル

- パレート解となり得るかつ更新対象とする解のラベル

以下に無向グラフに対するアルゴリズムを記載する.

記号

k：最適化目的の数

$v \in V$，$j = 1, \ldots, k$ に対して

l_{jv}：始点からノード v に到達したときに生じる第 j 番目の目的関数における総コスト

$e \in E$，$j = 1, \ldots, k$ に対して

e_{jw}：辺 e の第 j 番目の目的関数におけるコスト

L_T：更新対象とするラベル

L_P：更新対象としないラベル

アルゴリズム

入力：グラフ $G = (V, E)$，始点 $s \in V$，最適化目的の数 k，各辺の重みを返す関数 $w : E \to \mathbb{R}^k$

出力：s から全ての頂点への最短経路となるパレート解の集合

Step 1. $L_T \leftarrow (s, 0, \ldots, 0)$，$L_P \leftarrow \emptyset$

Step 2. $L_T = \{\emptyset\}$ となるまで以下の操作を行う.

Step 2-1. L_T の先頭にある $(\omega, l_{1\omega}, \ldots, l_{k\omega}) \in L_T$ を選択する.

Step 2-2. $L_T \leftarrow L_T \setminus \{(\omega, l_{1\omega}, \ldots, l_{k\omega})\}$

Step 2-3. 以下の条件を満たすとき, $L_P \leftarrow L_P \cup \{(\omega, l_{1\omega}, \ldots, l_{k\omega})\}$ とする.

- 任意の $(\omega^*, l_{1\omega^*}, \ldots, l_{k\omega^*}) \in L_P$ に $(\omega, l_{1\omega}, \ldots, l_{k\omega})$ が支配されない.
- 任意の $(\omega^*, l_{1\omega^*}, \ldots, l_{k\omega^*}) \in L_P$ と $(\omega, l_{1\omega}, \ldots, l_{k\omega})$ における全ての目的関数値が同値でない.

Step 2-4. 頂点 ω に辺 e によって接続している頂点 u に対して以下の操作を行う.

Step 2-4-1. 辺 e の重みベクトルを $\vec{e} = (e_{1w}, \ldots, e_{kw})$ とし，$(\omega', l_{1\omega'}, \ldots, l_{k\omega'}) \leftarrow (\omega', l_{1\omega} + e_{1w}, \ldots, l_{k\omega} + e_{kw})$ とする．

Step 2-4-2. 以下の条件を満たすとき，$L_T \leftarrow L_T \cup \{(\omega', l_{1\omega'}, \ldots, l_{k\omega'})\}$ とする．

- 任意の $(\omega^*, l_{1\omega^*}, \ldots, l_{k\omega^*}) \in L_P \cup L_T$ に $(\omega', l_{1\omega'}, \ldots, l_{k\omega'})$ が支配されない．
- 任意の $(\omega^*, l_{1\omega^*}, \ldots, l_{k\omega^*}) \in L_P \cup L_T$ と $(\omega', l_{1\omega'}, \ldots, l_{k\omega'})$ における全ての目的関数値が同値でない．

Step 2-4-3. 任意の $(\omega'', l_{1\omega''}, \ldots, l_{k\omega''}) \in L_T$ に対して $(\omega', l_{1\omega'}, \ldots, l_{k\omega'})$ が $(\omega'', l_{1\omega''}, \ldots, l_{k\omega''})$ を支配しているとき，$L_T \leftarrow L_T \backslash \{(\omega'', l_{1\omega''}, \ldots, l_{k\omega''})\}$ とする．

Step 3. 全てのパレート解を出力

支配

解 x, y が以下の条件を満たすとき，x は y を支配する．

- $\forall i \in \{1, \ldots, k\}, f_i(x) \leq f_i(y)$
- $\exists i \in \{1, \ldots, k\}, f_i(x) < f_i(y)$
- k：最適化目的の数，f：目的関数

3.3 ラベル付けアルゴリズムによる解法

ラベル付けアルゴリズムとは頂点数の数だけラベルを用意し，各頂点に対する解を分けて記憶するアルゴリズムである．各頂点ごとに記憶することでパレート解の判定の際，毎回どの頂点の path なのかを判定せずに比較対象を選択することができる．また，探索方法は未更新の頂点集合を用意し，未更新の頂点がなくなるまで全辺を緩めて更新する．以下に無向グラフに対するアルゴリズムを記載する．

記号

k： 最適化目的の数

$v \in V$ に対して

L_v：頂点 v に対するラベル

$v \in V$，$j = 1, \ldots, k$ に対して

l_{jv}：始点からノード v に到達したときに生じる第 j 番目の目的関数における総コスト

$e \in E$，$j = 1, \ldots, k$ に対して

e_{jw}：辺 e の第 j 番目の目的関数におけるコスト

X：更新対象とする頂点集合

アルゴリズム

入力：グラフ $G = (V, E)$，始点 $s \in V$，最適化目的の数 k，各辺の重みを返す関数 $w : E \to \mathbb{R}^k$

出力：s から全ての頂点への最短経路となるパレート解の集合

Step 1. $\forall v \in V, L_v \leftarrow \emptyset$，$L_s \leftarrow (s, 0, \ldots, 0)$，$X \leftarrow s$

Step 2. $X = \{\emptyset\}$ となるまで以下の操作を行う．

Step 2-1. $v \in X$ となる頂点 v を選択する．

Step 2-2. $X \leftarrow X \setminus \{v\}$

Step 2-3. $e = e \in E \mid v, u \in e$ となる e に対して以下の操作を行う．

Step 2-3-1. 頂点 v に対する全ての path $(v', l_{1v'}, \ldots, l_{kv'}) \in L_v$ に対して以下の操作を行う．

Step 2-3-1-1. 辺 e の重みベクトルを $\vec{e} = (e_{1w}, \ldots, e_{kw})$ とし，$(u', l_{1u'}, \ldots, l_{ku'}) \leftarrow (u', l_{1v} + e_{1w}, \ldots, l_{kv} + e_{kw})$ とする．

Step 2-3-1-2. 以下の条件を満たすとき，$L_u \leftarrow L_u \cup \{(u', l_{1u'}, \ldots, l_{ku'})\}$，$X \leftarrow X \cup \{u\}$ とする．

- 任意の $(u^*, l_{1u^*}, \ldots, l_{ku^*}) \in L_u$ に $(u', l_{1u'}, \ldots, l_{ku'})$ が支配されない．
- 任意の $(u^*, l_{1u^*}, \ldots, l_{ku^*}) \in L_u$ と $(u', l_{1u'}, \ldots, l_{ku'})$ における全ての目的関数値が同値でない．

Step 2-3-1-3. 任意の $(u'', l_{1u''}, \ldots, l_{ku''}) \in L_u$ に対して $(u', l_{1u'}, \ldots, l_{ku'})$ が $(u'', l_{1u''}, \ldots, l_{ku''})$ を支配しているとき，$L_u \leftarrow L_u \setminus \{(u'', l_{1u''}, \ldots, l_{ku''})\}$ とする．

Step 3. 全てのパレート解を出力

支配

解 x, y が以下の条件を満たすとき，x は y を支配する．

- $\forall i \in \{1, \ldots, k\}, f_i(x) \leq f_i(y)$
- $\exists i \in \{1, \ldots, k\}, f_i(x) < f_i(y)$
- k：最適化目的の数，f：目的関数

第4章　解法の提案と実験的評価

この章では，提案解法の紹介と実装における工夫，本研究に対する成果を述べる．

4.1　問題の分析

本研究では以下の理由に基づき，有向完全グラフをインスタンスとする．また，提案解法 1 と提案解法 2 のインスタンスは非負数とする．

- 多目的最短経路問題において，完全グラフがもっともパレート最適解の数が多く計算が難しい．
- 負の値を考慮するとき，有向グラフをインスタンスとする．

以下にそれぞれの詳細を記載する．

4.2　非負数の問題に対する解法の提案

4.2.1　提案解法 1

記号

k： 最適化目的の数

$v \in V$ に対して

L_v： 頂点 v に対するラベル

$M(L_v)$： 頂点 v に対するラベルにおいて，更新対象とその他の境界となる解（$M(L_v)$ が L_v の i 番目の解のとき，i+1 番目以降の解は更新対象とする）

$v \in V$，$j = 1, \ldots, k$ に対して

l_{jv}： 始点からノード v に到達したときに生じる第 j 番目の目的関数における総コスト

$e \in E$，$j = 1, \ldots, k$ に対して

e_{jw}：辺 e の第 j 番目の目的関数におけるコスト

アルゴリズム

入力：グラフ $G = (V, E)$，始点 $s \in V$，最適化目的の数 k，各辺の重みを返す関数 $w : E \to \mathbb{R}^k$

出力：s から全ての頂点への最短経路となるパレート解の集合

Step 1. $\forall v \in V, L_v \leftarrow \emptyset$，$L_s \leftarrow (s, 0, \ldots, 0)$

Step 2. 更新ができなくなるまで以下の操作を行う．

Step 2-1. $\forall v \in V$ となる頂点 v に対して以下の操作を行う．

Step 2-1-1. $\forall u \in V$ となる頂点 u に対して以下の操作を行う．

Step 2-1-1-1. L_v から L_u に対しての更新を行う．

Step 2-1-2. $M(L_v)$ を L_v の最後の解とする．

Step 3. 全てのパレート解を出力

L_v から L_u に対しての更新

Step 1. L_v，L_u，頂点 v から頂点 u への辺 e を受け取る．

Step 2. $M(L_v)$ より後にある全ての解 $(v', l_{1v'}, \ldots, l_{kv'}) \in L_v$ について以下の操作を行う．

Step 2-1. 辺 e の重みベクトルを $\vec{e} = (e_{1w}, \ldots, e_{kw})$ とし，$(u', l_{1u'}, \ldots, l_{ku'}) \leftarrow (u', l_{1v} + e_{1w}, \ldots, l_{kv} + e_{kw})$ とする．

Step 2-2. 以下の条件を満たすとき，$L_u \leftarrow L_u \cup \{(u', l_{1u'}, \ldots, l_{ku'})\}$ とする．

- 任意の $(u^*, l_{1u^*}, \ldots, l_{ku^*}) \in L_u$ に $(u', l_{1u'}, \ldots, l_{ku'})$ が支配されない．
- 任意の $(u^*, l_{1u^*}, \ldots, l_{ku^*}) \in L_u$ と $(u', l_{1u'}, \ldots, l_{ku'})$ における全ての目的関数値が同値でない．

Step 2-3. 任意の $(u'', l_{1u''}, \ldots, l_{ku''}) \in L_u$ に対して $(u', l_{1u'}, \ldots, l_{ku'})$ が $(u'', l_{1u''}, \ldots, l_{ku''})$ を支配しているとき $L_u \leftarrow L_u \setminus \{(u'', l_{1u''}, \ldots, l_{ku''})\}$ とする．

支配

解 x, y が以下の条件を満たすとき，x は y を支配する．

- $\forall i \in \{1, \ldots, k\}, f_i(x) \leq f_i(y)$
- $\exists i \in \{1, \ldots, k\}, f_i(x) < f_i(y)$
- k：最適化目的の数，f：目的関数

4.2.2 提案解法２

記号

k： 最適化目的の数

$v \in V$ に対して

L_{vx}： 頂点 v に対する，更新対象としないラベル

L_{vy}： 頂点 v に対する，更新対象とするラベル

L_{vz}： 頂点 v に対する，次回探索で更新対象とするラベル

$v \in V$，$j = 1, \ldots, k$ に対して

l_{jv}： 始点からノード v に到達したときに生じる第 j 番目の目的関数における総コスト

$e \in E$，$j = 1, \ldots, k$ に対して

e_{jw}： 辺 e の第 j 番目の目的関数におけるコスト

アルゴリズム

入力： グラフ $G = (V, E)$，始点 $s \in V$，最適化目的の数 k，各辺の重みを返す関数 $w : E \to \mathbb{R}^k$

出力： s から全ての頂点への最短経路となるパレート解の集合

Step 1. $\forall v \in V, L_v \leftarrow \emptyset$，$L_{sy} \leftarrow (s, 0, \ldots, 0)$

Step 2. 更新ができなくなるまで以下の操作を行う．

Step 2-1. $\forall v \in V$ となる頂点 v に対して以下の操作を行う．

Step 2-1-1. $\forall u \in V$ となる頂点 u に対して以下の操作を行う．

Step 2-1-1-1. L_{vy} から L_u に対しての更新を行う．

Step 2-2. $\forall v \in V$ に対して L_{vx}, L_{vy}, L_{vz} の更新を行う．

Step 3. 全てのパレート解を出力

L_{vy} から L_u に対しての更新

Step 1. L_{vy}，L_u，頂点 v から頂点 u への辺 e を受け取る．

Step 2. $\forall(v', l_{1v'}, \ldots, l_{kv'}), (v', l_{1v'}, \ldots, l_{kv'}) \in L_{vy}$ について以下の操作を行う．

Step 2-1. 辺 e の重みベクトルを $\vec{e} = (e_{1w}, \ldots, e_{kw})$ とし，$(u', l_{1u'}, \ldots, l_{ku'}) \leftarrow (u', l_{1v} + e_{1w}, \ldots, l_{kv} + e_{kw})$ とする．

Step 2-2. 以下の条件を満たすとき，$L_{uz} \leftarrow L_{uz} \cup \{(u', l_{1u'}, \ldots, l_{ku'})\}$ とする．

- 任意の $(u^*, l_{1u^*}, \ldots, l_{ku^*}) \in \{L_{ux} \cup L_{uy} \cup L_{uz}\}$ に $(u', l_{1u'}, \ldots, l_{ku'})$ が支配されない．
- 任意の $(u^*, l_{1u^*}, \ldots, l_{ku^*}) \in \{L_{ux} \cup L_{uy} \cup L_{uz}\}$ と $(u', l_{1u'}, \ldots, l_{ku'})$ における全ての目的関数値が同値でない．

Step 2-3. 任意の $(u'', l_{1u''}, \ldots, l_{ku''}) \in L_{ux}$ に対して $(u', l_{1u'}, \ldots, l_{ku'})$ が $(u'', l_{1u''}, \ldots, l_{ku''})$ を支配しているとき $L_{ux} \leftarrow L_{ux} \backslash \{(u'', l_{1u''}, \ldots, l_{ku''})\}$ とする．

Step 2-4. 任意の $(u'', l_{1u''}, \ldots, l_{ku''}) \in L_{uy}$ に対して $(u', l_{1u'}, \ldots, l_{ku'})$ が $(u'', l_{1u''}, \ldots, l_{ku''})$ を支配しているとき $L_{uy} \leftarrow L_{uy} \backslash \{(u'', l_{1u''}, \ldots, l_{ku''})\}$ とする．

Step 2-5. 任意の $(u'', l_{1u''}, \ldots, l_{ku''}) \in L_{uz}$ に対して $(u', l_{1u'}, \ldots, l_{ku'})$ が $(u'', l_{1u''}, \ldots, l_{ku''})$ を支配しているとき $L_{uz} \leftarrow L_{uz} \backslash \{(u'', l_{1u''}, \ldots, l_{ku''})\}$ とする．

L_{vx}, L_{vy}, L_{vz} の更新

Step 1. L_{vx}, L_{vy}, L_{vz} を受け取る．

Step 2. $L_{vx} = L_{vx} \cup L_{vy}$ とし，$L_{vy} = \{\emptyset\}$ とする．

Step 3. $L_{vy} \leftarrow L_{vz}$ とし，$L_{vz} = \{\emptyset\}$ とする．

支配

解 x, y が以下の条件を満たすとき，x は y を支配する．

- $\forall i \in \{1, \ldots, k\}, f_i(x) \leq f_i(y)$
- $\exists i \in \{1, \ldots, k\}, f_i(x) < f_i(y)$
- k：最適化目的の数，f：目的関数

4.3 負の値の考慮した問題に対する解法の提案

提案解法 2 を基に負の値の考慮を行う．

4.4 実装における工夫

4.5 提案解法の評価と分析

第5章　結論

5.1　研究成果

5.2　今後の課題

謝辞

本研究を進めるにあたり，指導教員の宋 少秋教授には研究に対する助言や熱心な指導をしていただきましたことを心から感謝いたします．またゼミや日常で多くの知識や示唆をいただいた研究室の先輩，同期の方々に感謝いたします．

2019 年 1 月 31 日 中野　壱帥

卒業論文　多目的最短経路問題における
動的計画法に基づいた拡張ベルマンフォード法の提案

中野　壱帥