Vol.2022-AL-187 No.4
2022/3/14

最大次数3のグラフにおける辞書式深さ優先探索から極大辞書式最小パスへのサイズ保存対数領域帰着

細川 秀樹[1,a)]　江口 僚太[1]　大舘 陽太[2]　泉 泰介[3]

概要：
辞書順式深さ優先探索 (Lexicographic Depth-First Search: Lex-DFS) 問題とは, 深さ優先探索問題において, 隣接リストに存在する最初の未訪問の頂点へフォワードする制約を課した深さ優先探索である．また極大辞書式最小パス (Lexicographically minimal maximal path: LMMP) とは，Lex-DFS において最初のバックトラックが発生するまでのパスを発見する問題である．本論文では，最大次数 3 の無向グラフにおける Lex-DFS から LMMP への対数領域サイズ保存帰着を示す．本帰着と既知の結果を組み合わせることで，疎なグラフに対する劣線形領域 Lex-DFS アルゴリズムを実現するためには最大次数 3 の無向グラフに対する劣線形領域の LMMP アルゴリズムを実現すれば十分であることが結論づけられる．

キーワード：辞書式深さ優先探索，極大辞書式最小パス，領域複雑性，対数領域帰着

A Size-Preserving Logspace Reduction from Lexicographic Depth-First Search to Lexicographic Min-Max Path in Graphs of Maximum Degree Three

Hosokawa Hideki[1,a)]　Eguchi Ryota[1]　Otachi Yota[2]　Izumi Taisuke[3]

Abstract: Lexicographic DFS (Lex-DFS) is a popular variant of DFS, which requires the search head always moves to the first undiscovered neighbor in the adjacency list of the current vertex (as long as it exists). The lexicographic min-max path problem (LMMP) is a weaker variant of Lex-DFS, which requires us to output the prefix of Lex-DFS ordering up to the vertex where the first backtrack occurs. In this paper, we presents a size-preserving logspace reduction from Lex-DFS to LMMP for undirected graphs of maximum degree three. Combining this result with some prior work, one can conclude that it suffices to develop a sublinear-space LMMP algorithm for obtaining a sublinear-space Lex-DFS for sparse graphs.

Keywords: lexicographic depth first search, lexicographic min-max path, space complexity, logspace reduction

1. はじめに

深さ優先探索 (DFS) は基本的なグラフ探索アルゴリズムの一つであり，数多くの応用が存在する．辞書順式深さ優先探索 (Lex-DFS) は DFS の部分問題の一つであり，探索の中で次に進む未訪問頂点として隣接リスト中で先頭に存在する頂点が必ず選ばれるという制約を満たすものを言う．

[1] 名古屋工業大学
Nagoya Institute of Technology, Gokiso-cho, Showa-ku, Nagoya, Aichi, 466-8555,Japan
[2] 名古屋大学
Nagoya University,Furo-cho,Tikusa-ku, Nagoya, Aichi 464-8603,Japan
[3] 大阪大学
Osaka University, 1-1 Yamadaoka, Suita, Osaka, 565-0871,Japan
[a)] cmc14120@ict.nitech.ac.jp

近年，DFS を含む基本的なグラフの問題の空間複雑性に対して，各種の自明な上界より効率的なアルゴリズムの開発が注目を集めている．これら研究の動機として，入力データが非常に大きいビッグデータや, 使用できるメモリが少ない極小デバイス上で動作するアルゴリズム等，線形空間以上を必要とするアルゴリズムが必ずしも効率的と言い難くなっていることなどが挙げられる．本研究は Lex-DFS の空間複雑性に着目する．本論文において，Lex-DFS 問題を Lex-DFS アルゴリズムでグラフを探索した場合の訪問順序を順に出力する問題と定義し，その空間複雑性を古典的な読み取り専用モデル [1] において必要とされる作動領域メモリの量と定義する．

本研究では，この問題に対して，最大次数 3 のグラフにおける極大辞書式最小パスへのサイズ保存対数領域帰着を提案する．極大辞書式最小パスとは，Lex-DFS 問題の部分問題の一つであり，最初にバックトラックを行うまでの Lex-DFS の訪問順を出力する問題である．形式的には，本研究では以下の定理を示す．

定理 1.1. 最大次数 3 の任意のグラフにおいて, $O(n/f(n))$-ビットメモリを用いる多項式時間極大辞書式最小パスを解くアルゴリズムが存在すると仮定する．このとき，任意の最大次数 3 のグラフにおいて $O(n/f(n))$-ビットメモリを用いて Lex-DFS を解くアルゴリズムが存在する．

1.1 関連研究

アルゴリズムの空間複雑性に関する計算量理論的なアプローチについては，対数領域帰着を用いた P-完全性の概念が代表的である．Reif による証明 [2], および Anderson と Meyr による証明 [3] によって，Lex-DFS 問題，ならびに極大辞書式最小パスはいずれも対数領域帰着のもとで P-完全であることが示されている．これは，$P \neq L$ が成立する場合，$O(\log n)$ ビットの動作領域を用いてこれらの不可能であることを示している．ここで，n は入力グラフの頂点数を表す．P-完全性の定義より，この事実は Lex-DFS から極大辞書式最小パスへの対数領域帰着が存在することを意味する．ただし，この帰着は入力インスタンスのサイズを漸近的に保存する帰着（サイズ保存帰着）ではない．すなわち，上記の帰着を用いた場合，頂点数 $O(n)$ の Lex-DFS 問題のインスタンスは頂点数 $O(n^c)$（$c > 1$) への極大辞書式最小パスのインスタンスへと変換されるため，本研究で示す主定理を導くことができない．Lex-DFS の上界の結果として，多項式時間で $o(n \log n)$ ビット領域[*1]を用いた (Lex-)DFS アルゴリズムが提案されている，代表的な結果として，Asano らによる多項式時間，$n + o(n)$ ビットメモリのアルゴリズムなどがある [4]．しかしながら，$o(n)$-ビットメモリ（もしくは，$c < 1$ に対して cn-ビットメモリ）を用いたアルゴリズムは依然として知られておらず，そのようなアルゴリズムが実現可能であるかどうかは興味深い未解決問題となっている．この問題に対して，ごく最近に Izumi [5] による以下の定理が示されている．

定理 1.2.（Izumi [5]）最大次数 3 の任意のグラフにおいて, $O(n/f(n))$-ビットメモリを用いる多項式時間 Lex-DFS アルゴリズムが存在すると仮定する．このとき，m 辺からなる任意のグラフにおいて $O(m/f(m))$ ビットメモリを用いる多項式時間 Lex-DFS アルゴリズムが存在する．

定理 1.2 と本研究における定理を組み合わせると，疎なグラフにおいて $o(n)$-ビットメモリを用いて多項式時間で Lex-DFS を解くためには，最大次数 3 のグラフにおいて極大辞書式最小パスを発見するアルゴリズムさえ設計すれば十分であることが分かる．なお，最大次数を 3 に制限した場合の Lex-DFS を計算するアルゴリズムについては，川端と泉 [6] による $0.98n + o(n)$-ビットメモリの多項式時間アルゴリズムが知られている．しかしながら，$o(n)$-ビットメモリを用いた多項式時間 Lex-DFS アルゴリズムが存在するかどうかは，依然として未解決のままである．

1.2 論文の構成

本論文の構成について述べる．第 2 節ではモデルやアルゴリズムの諸定義を行い，第 3 節では本論文が提案する帰着方法の詳細を述べたのち，第 4 節においてその正当性を証明する．第 5 節おいてまとめと今後の課題を述べる．

2. 諸定義

2.1 モデルと表記

本論文で用いる計算モデルは読み取り専用モデル [1] を用いる．読み取り専用モデルにおいて，メモリは読み出し専用の入力メモリ, 書き込み専用の出力メモリ, アルゴリズムが使用する作業メモリの三つからなる. アルゴリズムの空間複雑性は作業メモリの量で測定する．メモリのアクセスは $O(\log n)$ ビットの単位文字へとランダムアクセスが可能なモデルを用いる．本研究では最大次数 3 の無向グラフ G を入力として考えるが，議論の簡便のため，G はすべての頂点の次数が 3 である，すなわち 3-正則グラフであることを仮定する．また，帰着の都合上，入力グラフ G は無向辺 $\{u, v\}$ の代わりに 2 本の有向辺 $(u, v), (v, u)$ を持つ有向グラフとみなす．任意のグラフ H に対して，以降は指定がなければその頂点集合と辺集合を V_H，E_H で表す．入力グラフは，隣接リスト A_G の形で入力メモリに格納されているものとする．ここで，$V_G = [0, n-1]$ とする，すなわち，各頂点の ID は範囲 $[0, n-1]$ 内の整数を取る．$N_G(v) \subseteq V_G$ によって頂点 v の隣接頂点の集合を表すものとし，またグラフ G における頂点 v の隣接リストを $A_{G,v}$ で表す．また，$A_{G,v}$ 内の i-番目の隣接頂点を $A_{G,v}[i]$ で表すものとする（添え字 i は 1 から始まるものとする）．ある

[*1] $O(n \log n)$ ビット領域は Lex-DFS の自明な上界である．

有向辺 $e = (u, v) \in E_G$ について，$A_{G,u}[i] = v$ となるような値 i を u における v のインデックス，あるいは辺 e の u におけるポート番号と呼ぶ.

2.2 辞書式順深さ優先探索 (Lex-DFS)

深さ優先探索 (DFS) とは探索ヘッドが指す頂点 v_h について、未訪問の隣接頂点がある場合は, その頂点に進み, 未訪問の隣接頂点がない場合は,v_h に初めて訪問した時の直前の頂点 (親頂点) へと戻る操作を可能な限り繰り返すものである．もし v_h の隣接頂点で未訪問のものが複数ある場合は，そのうちの一つを自由に選ぶ．Lex-DFS とは，探索中の頂点 u において未訪問頂点が（複数）存在する場合，隣接未訪問頂点のうち u におけるインデックスが最小のものを選び次訪問ノードとする制約を課した DFS である．

本稿を通して，G を入力グラフとし，Lex-DFS の開始頂点を $s \in V_G$ で表すものとする．また探索中に隣接頂点で未訪問のものがあるとき, その頂点に進むことをフォワード動作と呼び，全ての隣接頂点が既訪問になった時，親頂点へ戻る操作をバックトラック動作と呼ぶ.

通常の Lex-DFS の実現においては，各頂点に対して既訪問, 未訪問のラベルを用意して，隣接する未訪問頂点（のうち最小インデックスのもの）を順次訪問するという形をとるが，今回は帰着手法の説明の都合上，頂点への訪問済みラベルの代わりに，各辺についてブロック，未ブロックの 2 つの状態を用意する．具体的には，アルゴリズムの初期状態において全辺は未ブロックの状態であり，Lex-DFS アルゴリズムが頂点 v を訪問した時点で，v へと向かう辺をすべてブロック状態とする．これにより頂点の既訪問, 未訪問の判定を辺のブロックの有無によって確認を行うことが出来る．アルゴリズムは初めに初期頂点 s を定め,s に向かうフォワード辺をブロックする．その後, 隣接頂点へ向かう辺をポート番号の順に 1 から確認して，ブロックされていないものが見つかればそれをフォワード辺として辺の終点頂点 v へ進む．v を訪問した時点で,，訪問頂点 v に入ってくる辺をすべてブロックする．訪問頂点においてブロックされていないフォワード辺が存在しない無い場合は, 現在の頂点について全ての隣接頂点が既訪問になっていることになる．その後, 現在の頂点の親頂点があるなら, 親頂点へバックトラックする．もし親頂点がない場合は始点の隣接頂点が全て訪問済になり, 探索が終了する.

2.3 極大辞書式最小パス (LMMP)

本論文において極大辞書式最小パス（Lexicography Minimum Maximal Path:LMMP）とは, 入力として無向グラフ $G = (V_G, E_G)$ と初期頂点 $sinV_G$ を始点として，隣接リスト中でインデックスが最小となる未訪問の頂点を可能な限り移動したときに訪問した頂点の系列を出力する問題とする．定義より明らかに, LMMP の出力は入力グラフ G における s を始点とした単純パスをなる．このパスを LMMP パスと呼ぶ.

3. 提案する帰着

3.1 帰着の概要

Lex-DFS から LMMP への問題の帰着の基本的な方針は以下の通りである．入力グラフ G を LMMP のアルゴリズムを実行するグラフ G' へと変換し，G' 上で LMMP のアルゴリズムを実行する．この実行が出力する LMMP の頂点系列をストリーム的に G 上の Lex-DFS の頂点系列へと変換することで，帰着を達成する．提案する帰着は，入力グラフ G の最大次数が 3 であると仮定して，G' として頂点数 $O(n)$，辺数 $O(n)$ であるグラフを作成する．$o(n)$ ビットメモリでの帰着を行うために，変換後のグラフ G' をオンメモリで保存するのではなく，G' の各頂点への隣接リストへのアクセスを (読み出し専用メモリに格納されている G の情報を用いて) 模倣するアルゴリズムを設計することにより実現する．形式的には，入力グラフ G を G' に変換する関数 f と，G' 上の LMMP の実行により得られる頂点列を G 上の Lex-DFS の出力に変換する関数 g，及び入力グラフ G 上の初期頂点 s を変換後のグラフ G' 上の初期頂点 s' へと変換する関数 h により実現される．以下にその詳細を述べる．最大次数 3 となるグラフ G の集合族を $\mathcal{G}_3$ とし，関数 f を $f : \mathcal{G}_3 \to \mathcal{G}_3$ とする．また，入力グラフ G と $G' = f(G)$ について，関数 g を $g : V_{G'} \to V_G \cup \{\perp\}$ と定義し，以下の性質を満たすものとする.

> $L = \ell_0, \ell_1, \ldots, \ell_{N-1}$ を G' 上の LMMP の出力列とし， $L' = \ell'_{i_0}, \ell'_{i_1}, \ldots, \ell'_{i_{N'-1}}$ を $g(u_j) \neq \perp$ であるような L' の部分系列としたとき，$g(\ell'_{i_0}), g(\ell'_{i_1}), \ldots, g(\ell'_{i_{N'-1}})$ が G の Lex-DFS の系列になる.

上の性質において, 直感的には, $g(\ell_j) = \perp$ であることは, u_j に対応する G 上の頂点が存在しないことを表す．LMMP の出力から Lex-DFS の出力を計算するためには，G' 上の出力に対して関数 g を計算し，その（出力 $\perp$ 以外の）値を出力することが必要となる．さらに Lex-DFS における初期頂点 $s \in V_G$ を $V_{G'}$ 上の頂点 s' に写す関数 $h : V_G \to V_{G'}$ を定義する．Lex-DFS から LMMP に対する $O(\log n)$ ビットスペースによる帰着とは, 以下の性質を満たす関数 f, g, h を構成することである.

(1) G から $G' = f(G)$ は $O(\log n)$ ビットの作業領域で構成可能である．すなわち，グラフ G' 上の頂点 v' を入力として，頂点 v' の隣接リスト $A_{G',v'}$ を出力する多項式時間決定性アルゴリズム $\mathcal{A}_{conv}$ が存在し，その空間複雑性は $O(logn)$ ビットである.

(2) G' 上の LMMP の任意の出力 v' に対して，関数 g を計算するアルゴリズム $\mathcal{A}_{out}$ が存在し，その空間複雑

性は $O(\log n)$ ビットである．

(3) G 上の初期頂点 s に対して，G' 上の初期頂点 s' へと s を移す関数 h を計算するアルゴリズム $\mathcal{A}_{init}$ が存在し，その計算複雑性は $O(\log n)$ ビットである．

3.2 G' の構成

本節では，G' の詳細について説明する．帰着の基本的なアイデアは，G の Lex-DFS 木 T に対して，T をオイラーツアーにて巡回するような歩道が G' 中の LMMP のパスに対応するように，G 中の各頂点および辺を対応するガジェットへと置き換えていくことである．以降，$u \in V_G$ に対応するガジェットを H_u^V, 有向辺 $(u,v) \in E_G$ に対するガジェットを $H_{(u,v)}$ で表すとする．LMMP の探索パス (以降，LMMP パスと呼ぶ) が初めて頂点に H_u にに到達したときを考える．また，$A_{G,u} = (v_1, v_2, v_3)$ として，LMMP パスは辺ガジェット $H_{(v_i,u)}$ を通って H_u へと到達したとする．H_u に到達した LMMP パスはまず H_u 内に存在するいくつかの (i に応じた) 頂点を既訪問とすることで，u の Lex-DFS 木の親へのポインタ（ポート番号）i を記憶する．その後，LMMP パスは $H_{(v_1,u)}$，$H_{(v_2,u)}$，$H_{(v_3,u)}$ を順次訪問して，各ガジェット内部のいくつかの頂点を既訪問として辺をブロックする．その後，Lex-DFS における次のフォワード辺を $H_{(u,v_1)}$，$H_{(u,v_2)}$，$H_{(u,v_3)}$ を順次訪問することで探索する．フォワード辺 (u,v_j) $(j \in \{1,2,3\})$ を発見したときは次頂点 v_j のガジェット H_{v_j} へと訪問する．H_j よりバックトラックで戻ってきた，あるいは $H_{(u,v_j)}$ が既にブロックされているとき，LMMP パスは $j \in \{1,2\}$ ならば $H_{(v,v_{j+1})}$ へと向かう，$j = 3$ の時は，u の隣接頂点の探索はすべて終了しているので，バックトラックを実施する．LMMP パスは H_u へと再び進入して，初訪問時に記憶した i の値に対応する辺ガジェット $H_{(v_i,u)}$ へと向かう，辺ガジェット内にはバックトラック用のパスがフォワード用のパスとは別に存在し，LMMP はそのパスを通り v_i へと戻る．次節以降は，頂点ガジェットおよび辺ガジェットの定義の述べた後，G' においてそれらのガジェットがどのように相互接続されるかを説明する．

3.2.1 頂点ガジェット

頂点 $v \in V_G$ に対するガジェット H_v を図 1 に示す．H_v の頂点集合 V_{H_v} は，以下の 3 つの頂点群に分かれる．

- 書き込み頂点：$u^v_{win(1)}$，$u^v_{win(2)}$，$u^v_{win(3)}$，u^v_{wout} の 4 頂点．値 i をガジェットに記憶させるときに通過する頂点であり，$u^v_(win(i)$ から進入したした LMMP パスが u^v_{wout} を通ってガジェット外へと出ていくことで値 i が記憶される (図 2 における実線).
- 読み出し頂点：u^v_{rin}，$u^v_{rout(1)}$，$u^v_{rout(2)}$，$u^v_{rout(3)}$ の 4 頂点．頂点 u^v_{rin} へと到達した LMMP パスは，事前に記憶された値 i に対応した頂点 $u^v_{rout(i)}$ より外へと出ていく (図 2 における点線).
- 内部頂点：u^v_{-1}，u^v_{-2}，u^v_{-3} の 3 頂点．複数のパスの合流・分岐のために導入されている頂点群

形式的には，形式的には，辺集合 $E_{H_{vw}}$ および各頂点の隣接リストは以下のように定義される．

- $A_{G',u^v_{rout(1)}} = (u^v_{-1}, \cdot, u^v_{rin})$
- $A_{G',u^v_{-1}} = (u^v_{-3}, u^v_{win(1)}, u^v_{rout(1)})$
- $A_{G',u^v_{win(1)}} = (u^v_{-1}, u^v_{rout(2)}, \cdot)$
- $A_{G',u^v_{rout(2)}} = (u^v_{-2}, \cdot, u^v_{win(1)})$
- $A_{G',u^v_{-2}} = (u^v_{-3}, u^v_{win(2)}, u^v_{rout(2)})$
- $A_{G',u^v_{win(2)}} = (u^v_{-2}, u^v_{rout(3)}, \cdot)$
- $A_{G',u^v_{-3}} = (u^v_{wout}, u^v_{-1}, u^v_{-2})$
- $A_{G',u^v_{wout}} = (\cdot, u^v_{-3}, u^v_{win(3)})$

上の記述において，各隣接リスト中に含まれる空欄 $(\cdot)$ は，当該箇所にガジェット外の頂点が入ることを意味する．どの頂点が入るか（すなわち，ガジェットの外に伸びる辺がどこにつながるか）については節 3.2.3 において示す．

3.2.2 辺ガジェット

有向辺 $(v,w) \in V_G$ に対するガジェット H_{vw} を図 3 に示す．頂点ガジェットの定義定義と同様に，頂点集合 $V_{H_{vw}}$ は，以下の 3 つの頂点群に分かれる．

- フォワード頂点：u^{vw}_{fin}，u^{vw}_{fout} の 2 頂点．Lex-DFS の実行における頂点 v から w へのフォワード動作に対応する形で，LMMP パスが u^{vw}_{fin} から進入して u^{vw}_{fout} から w の頂点ガジェットへと出ていく（図 4(1) のフォワード動作参照).
- ブロック頂点：u^{vw}_{bin}，u^{vw}_{bout} の 2 頂点．Lex-DFS の実行における辺 (v,w) のブロック操作に対応する形で，LMMP パスが u^{vw}_{bin} から進入して u^{vw}_{bout} からガジェット外へと出ていく（図 4 の辺ブロック動作参照).
- リターン頂点：u^{vw}_{rin}，u^{vw}_{rout} の 2 頂点．頂点 w から v へのバックトラック動作に対応する形で，LMMP パスが u^{vw}_{rin} から進入して u^{vw}_{rout} からガジェット外へと出ていく（図 4 の辺ブロック動作参照)．また，v から w へのフォワード動作を試みたときに，辺 (v,w) が既にブロック済みのとき，LMMP パスは u^{vw}_{fin} から進入して u^{vw}_{rout} から引き返していく （図 4 の引き返し動作参照)
- 内部頂点：u^v_{-1}，複数のパスの合流・分岐のために導入されている頂点.

形式的には，辺集合 $E_{H_{vw}}$ および各頂点の隣接リストは以下のように定義される．

- $A_{G',u^{vw}_{fin}} = (u^{vw}_{-1}, u^{vw}_{rout}, \cdot)$
- $A_{G',u^{vw}_{rout}} = (\cdot, u^{vw}_{fin}, u^{vw}_{rin})$
- $A_{G',u^{vw}_{-1}} = (u^{vw}_{fout}, u^{vw}_{bout}, u^{vw}_{fin})$
- $A_{G',u^{vw}_{bout}} = (\cdot, u^{vw}_{-1}, u^{vw}_{bin})$
- $A_{G',u^{vw}_{bin}} = (u^{vw}_{fout}, u^{vw}_{bout}, \cdot)$
- $A_{G',u^{vw}_{fout}} = (u^{vw}_{-1}, \cdot, u^{vw}_{bin})$

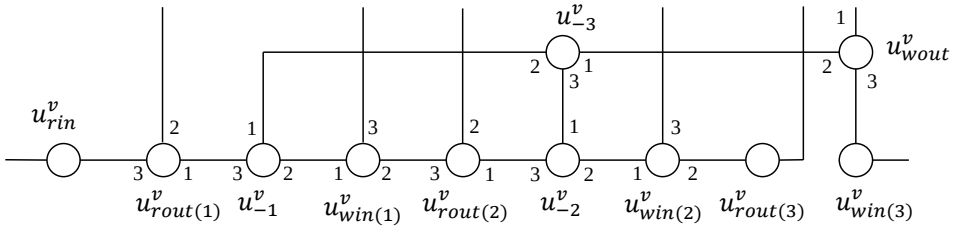

図 1 頂点ガジェット H_v のグラフ

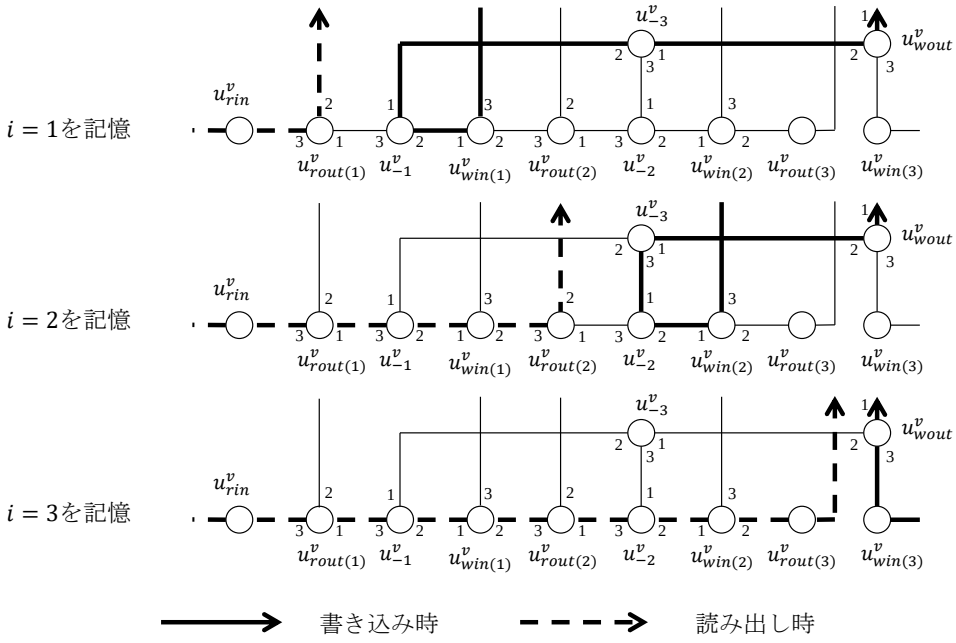

図 2 頂点ガジェット H_v における書き込みと読み出し

3.2.3 ガジェット間の接続

頂点 x の隣接頂点リストを $A_{G,x} = (y_1, y_2, y_3)$ として，有向辺 e_i を $e_i = (y_1, x)$ とする．また，辺 e の逆向き辺を $\bar{e}$ で表すとする．x および $e_1, e_2, e_3, \bar{e}_1, \bar{e}_2, \bar{e}_3$ に対応するガジェットは以下のように接続される．

- すべての $i \in \{1,2,3\}$ について，$u^{e_i}_{fout}$ と $u^x_{win(i)}$ を接続する．
- u^x_{wout} と $u^{e_1}_{bin}$ を接続する．また，$i \in \{1,2\}$ について，$u^{e_i}_{bout}$ と $u^{e_{i+1}}_{bin}$ を接続する．
- $u^{e_3}_{bout}$ と $u^{\bar{e}_1}_{fin}$ を接続する．また，$i \in \{1,2\}$ について，$u^{\bar{e}_i}_{rout}$ と $u^{e_{i+1}}_{fin}$ を接続する．
- $u^{\bar{e}_3}_{rout}$ と u^x_{rin} を接続する．
- すべての $i \in \{1,2,3\}$ について，$u^x_{rout(i)}$ と $u^{e_i}_{rin}$ を接続する．

ガジェット内の各頂点 x は高々 1 本のガジェット外へと伸びる辺を持ち，また x に接続するガジェット内の辺の x については，そのポート番号は節 3.2.1 および節 3.2.2 において定められている，そのため上で意義されるガジェット間接続辺について，その両端点におけるポート番号は一意に決まる．ガジェット間の接続関係を図 5 に示す．

3.2.4 関数 f，g，h の構成

各頂点 $v \in V_G$ に対する頂点ガジェット，各辺 $e \in E_G$ に対する辺ガジェットのいずれも，その大きさは定数であり，与えられた v または e に対応するガジェットはすべてオンメモリで校正が可能である．また，頂点ガジェット H_v から出ていくガジェット間の辺についても，その接続先を含めてすべての v および v の G における隣接リスト $A_{G,v}$ より列挙が可能である．よって，G' における頂点 $v' \in V_{G'}$ の隣接リスト $A_{G',v'}$ を模倣する関数 f は $O(\log n)$ ビットの作業領域で容易に模倣可能である．関数 g については，以下のように定める．

$$g(x) = \begin{cases} v & x = u^v_{wout} \text{ のとき} \\ \perp & \text{それ以外} \end{cases}$$

また．関数 h は $h(s) = u^s_{win(3)}$ と定義する．直感的には，LMMP パスが頂点 u^v_{wout} を通過したタイミングを，グラフ G において Lex-DFS が頂点 v を訪問したタイミングとみなす．

4. 正当性の証明

上述の構成におけるグラフ G' において LMMP を実行することを考える．各頂点 $v \in V_G$ およびその隣接リスト $A_{g,v} = (w_1, w_2, w_3)$ について，頂点ガジェット H_v，辺ガジェット $H_{w_1 v}$，$H_{w_2 v}$，$H_{w_3 v}$ とその間を接続する辺から

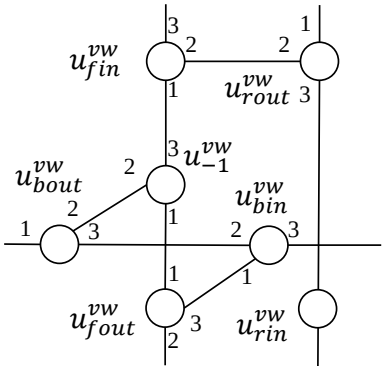

図 3 辺ガジェット $H_{v,w}$ のグラフ

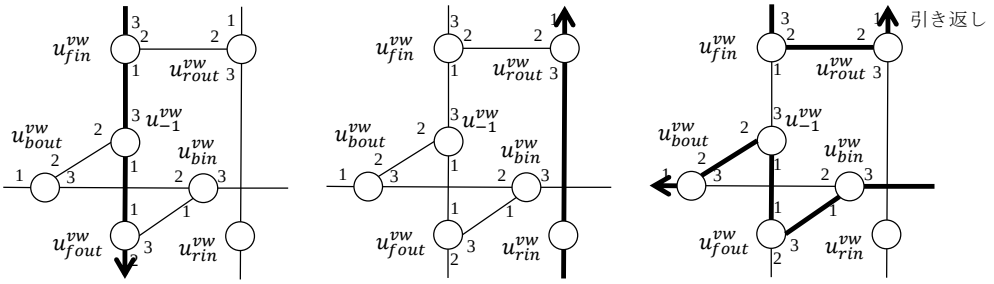

図 4 辺ガジェット $H_{v,w}$ におけるフォワード動作，バックトラック動作，辺ブロック

なる G' の部分グラフを v のユニットとよび，U_v で表す．$L' = \ell_0, \ell_1, \ldots, \ell_N$ を LMMP パスが通るグラフ G' の頂点系列とする．$\ell_i = v$ であるとき，i を LMMP パスにおける頂点 v の訪問時刻と呼び，LMMP パスは時刻 t に頂点 v を訪問したと呼ぶ．頂点 $v \in V_{G'}$ を LMMP パスが訪問した時刻を $t(v)$ で表すとする．LMMP パスが v を訪問しないときは $t(v) = \infty$ と定義する．時刻 t に LMMP パスが頂点 u^{vw}_{-1} を訪問済みの時，時刻 t において辺 (v, w) はブロックされていると呼ぶ．また，$t(U_v)$ を，ユニット U_v 中の頂点を最初に訪問する時刻とする．証明の骨子は以下の 2 つの補題である．

補題 4.1. $v \in V_G$ を任意の頂点として，$A_{G,v} = (w_1, w_2, w_3)$ とする．t を LMMP パスが U_v 中の頂点を訪問した最初の時刻として，$t' > t$ を t 以降で U_v に含まれない頂点を訪問する最初の時刻とする．$t = 0$，ある $i \in \{1, 2, 3\}$ について $\ell_{t(U_v)} = u^{w_i v}_{fin}$ であるならば，時刻 $t' - 1$ において以下の 2 つの条件が満たされる．

- (C1) LMMP パスは頂点 v へと向かう辺 (w_i, v) をすべてブロックしている．
- (C2) u^v_{-1}，u^v_{-2}，u^v_{-3} のうち，u^v_{-i} のみが訪問済み．
- (C3) u^v_{wout} が訪問済み．

から $t' - 1$ の間に，かつ

証明. 厳密な証明は略するが，補題の条件下では図 6 のようにLMMP パスが U_v 中を通過することよりほぼ明らかである (図は $i = 1$ の場合)． □

補題 4.2. $v_0, v_1, \ldots, v_{n-1}$ を G の Lex-DFS における出力系列とすると，以下が成り立つ．

- (S1) 任意の i について，$A_{G,v_i} = (w_1, w_2, w_3)$ とすると，$\ell_{t(U_{v_i})} \in \{u^{w_1 v_i}_{fin}, u^{w_2 v_i}_{fin}, u^{w_3 v_i}_{fin}\}$ である．
- (S2) $j < i$ であるような v_j，v_i について，$t(U_{v_j}) < t(U_{v_i})$.

証明. 補題の成立は，直感的には，図 7 に示すような形で LMMP パスの探索が進んでいくことからわかる（この図は $t(U_{w_2}) < t(U_{v_i})$ かつ $t(U_{w_1}) < t(U_(v_i)$ であり，Lex-DFS の実行において v_i へは w_2 からのフォワード動作により訪問しているとしている）．厳密な証明は i についての帰納法によるが，詳細は省略する． □

補題 4.2 より，$t(U_{v_0}) < t(U_{v_1}) < \cdots < t(U_{v_n - 1})$ が成立する．また，補題 4.1 の条件 (C3) より，時刻 $t(U_{v_i})$ と $t(U_{v_{i+1}})$ の間に LMMP パスは頂点 u^v_{wout} を訪問する．このことより，提案する帰着が正しく Lex-DFS を模倣することが示される．

5. まとめと今後の課題

本研究では，Lex-DFS の空間複雑性について研究し，その部分問題として最大次数 3 のグラフにおける Lex-DFS

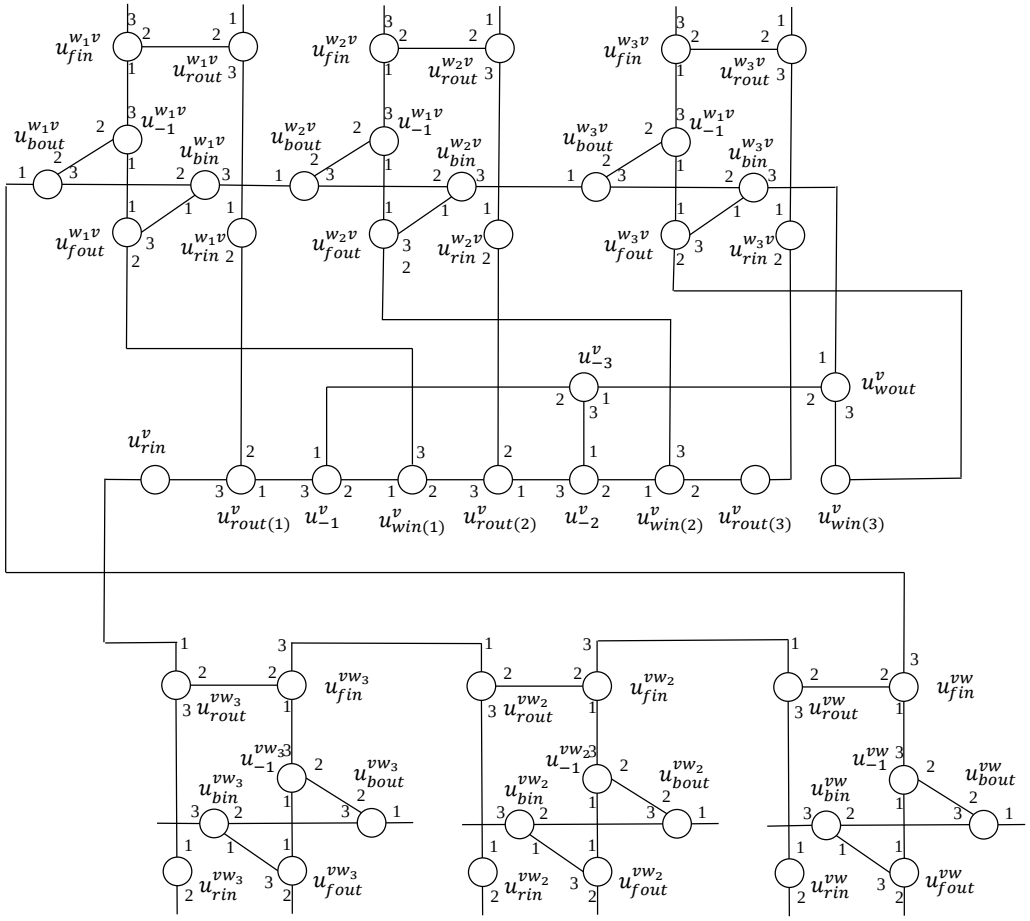

図 5 ガジェット間の接続

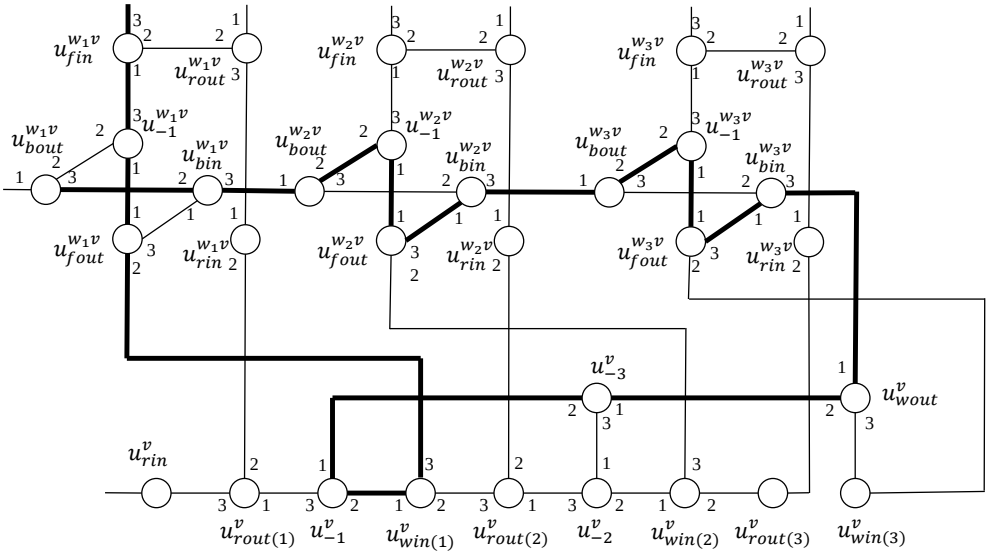

図 6 補題 4.1 の証明

から辞書式最小パスへとサイズ保存対数領域帰着を提案した．本研究の結果と先行研究の結果を合わせると，疎な（辺数が $O(m)$ であるような）グラフ上の Lex-DFS を $o(n)$-ビットメモリで実現するためには，最大字数 3 のグラフにおける辞書式最小パス問題に対する $o(n)$ ビットメモリアルゴリズムを設計さえすれば十分であることが示された．今後の課題として，LMMP の空間複雑性について考えることが挙げられる．

参考文献

[1] Greg N. Frederickson. Upper bounds for time-space trade-offs in sorting and selection. *Journal of Computer and System Sciences*, 34(1):19 – 26, 1987.

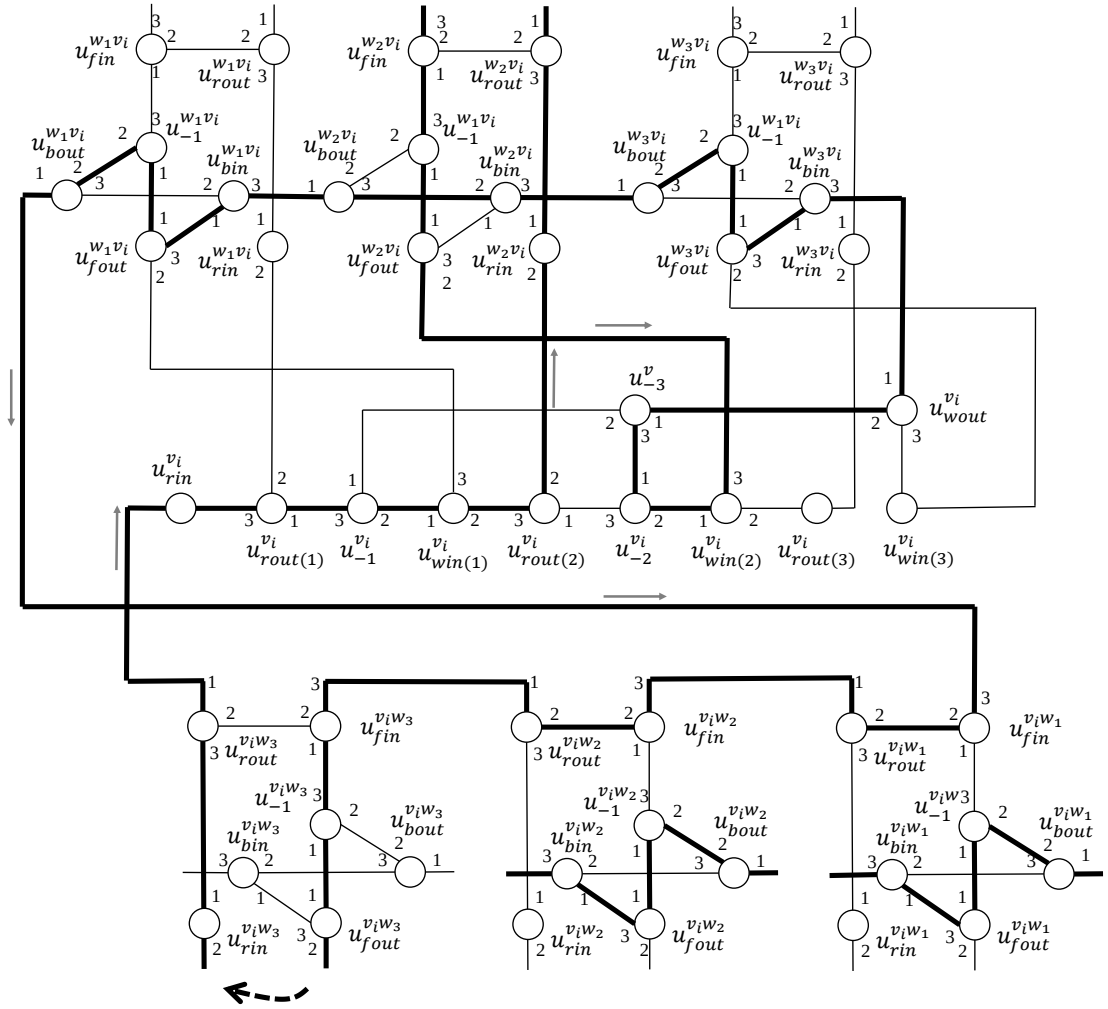

図 7 補題 4.2 の証明

[2] John H. Reif. Depth-first search is inherently sequential. *Information Processing Letters*, 20(5):229 – 234, 1985.

[3] Richard Anderson and Ernst W. Mayr. Parallelism and the maximal path problem. *Inf. Process. Lett.*, 24(2):121–126, 1987.

[4] Tetsuo Asano, Taisuke Izumi, Masashi Kiyomi, Matsuo Konagaya, Hirotaka Ono, Yota Otachi, Pascal Schweitzer, Jun Tarui, and Ryuhei Uehara. Depth-first search using $$o(n)$$bits. In *International Symposium on Algorithms and Computation (ISAAC)*, pages 553–564, 2014.

[5] Izumi Taisuke. On hardness of sublinear-space lex-dfs for graphs of maximum degree three. (2), sep 2019.

[6] 川端祐也 and 泉泰介. 最大次数 3 のグラフにおける n ビット未満の作業領域を用いた深さ優先探索. **研究報告アルゴリズム** *(AL)*, 2018(10):1–8, 2018.