頂点毎の通過単純閉路の数え上げに対するワンパスZDDアルゴリズム

大江　憂歌[1,a)]　来嶋　秀治

概要：本発表では $N \times N$ 格子グラフ上の各頂点について，その頂点を訪問する単純閉路の総数を求める方法について論じる．まず，その総数の上界が 2^{N^2} であることを示す．つぎに，zero-suppressed decision diagram (ZDD) を用いた数え上げアルゴリズムを提案する．既存の CyclePath では各頂点ごとに通過する単純閉路の総数を求めているが，提案手法では，閉路数のデータ構造を内蔵することにより，各頂点の通過単純閉路の個数がワンパスで求められる．

キーワード：グラフ理論, アルゴリズム,

One-pass ZDD Algorithm for Counting The Number of Visiting Simple Cycles for Each Vertex

Abstract: This paper is concerned with counting the number of simple cycles on the $N \times N$ grid, precisely for each vertex v, we count the number of simple cycles which visits v. Firstly, we prove that the number of simple cycles is at most 2^{N^2}. Next, we present an algorithm based on the zero-suppressed decision diagram (ZDD). While the existing algorithm "CyclePath" counts the number of cycles for each vertex, our algorithm outputs the numbers of cycles for all vertices at once.

Keywords: graph theory, algorithm

1. はじめに

グラフの列挙問題は社会的に重要なさまざまな実問題と関係している [1]. 例えばパス列挙は, 地理情報システムでは中心的な問題であるし, 有向グラフでの列挙は, 大規模システムの依存関係の解析や, フローチャートの解析, ナンバーリンクやスリザーリンクのパズル, 文字列集合から「しりとり」を完成させる問題など, 多くの応用がある.

単純経路とは同一点を 2 回は通らないという条件を満たす経路である. 格子グラフにおける単純経路の問題は湊離散構造処理系プロジェクト作成の動画 [8] によりお姉さん問題として現在広く知られている. この問題は格子グラフの左上端の点をスタート地点として, 右下端の点をゴール地点と設定し, 全ての単純経路の個数を求めるというものである. 格子数を増やし続けると単純な深さ優先探索などの数え上げでは数億年の計算時間と莫大なメモリを要する. しかし, 湊 [1] が考案した ZDD(Zero-Suppress decision diagram) により, 格子グラフの 2 点間の経路を高速に列挙することが可能となり, 単純な深さ優先探索では格子数が 10×10 で数億年かかる処理を数秒で計算することができる [1,8].

また, 上記のアルゴリズムを用いても, 格子グラフの経路列挙は格子数を増やし続けると莫大な時間が生じてしまう. 現在では, 格子数 26×26 までの格子グラフにおいて正確な経路の総数がわかっている. それに対し, 柴田ら [5] はマルコフ連鎖モンテカルロ法を用いることにより格子数 200 のグラフの単純経路数の近似値を求めている. ここで, 柴田らは始点と終点が外周にある場合の経路の総数の上界が 2^{N^2} であることを証明している. しかし, 始点と終点がグラフの内部にある場合経路の総数の上界を同様の方法で示すこ

[1] 九州大学
Kyushu University
[†1] 現在，九州大学
Presently with Kyushu University
[a)] yu_ka092701@me.com

とは困難であった.

本稿第 2 章では, サイクル経路の総数と, 始点と終点が隣接 2 点間である場合の経路の総数の上界が同様に 2^{N^2} であることを証明する. 第 3 章では単純経路の ZDD を構成し単純閉路の総数を求める方法を提案する. 既存の CyclePath アルゴリズム [1] では 1 つの点を通過する単純閉路の数え上げが, 指定頂点に対して, 提案アルゴリズムでは格子グラフにおける各頂点が自分自身を通過する単純閉路が格子グラフ内に何個存在するかを全頂点に対して, ワンパスで求める.

2. 格子グラフにおける経路数の上界

2.1 格子グラフにおける経路数の上界に関する先行研究

まず柴田らの研究 [5] により外周の頂点間を結ぶ経路数の上界がどのように求められたかの紹介を行う. 外周上の 2 点を選択した際の 1 本の経路に対して格子グラフの外側にその 2 点を含む線を引いて閉路を作り, 単連結な色塗りを行う.

格子の 1 つの色の塗り方に対して経路が 1 本に定まるため, 格子の色の塗り方は格子数が N^2 であることから 2^{N^2} である. 従って単純経路の総数の上界も同様に 2^{N^2} である.

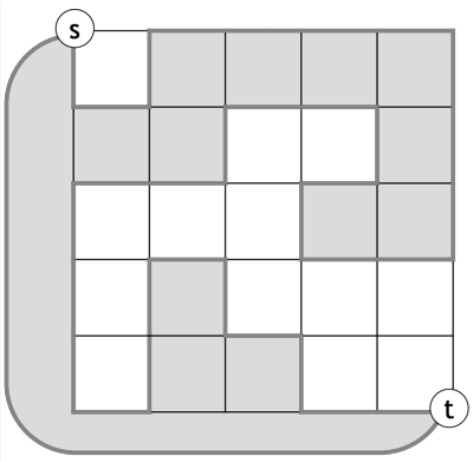

図 1: 色塗りによる証明
Fig. 1 Proof by coloring

2.2 隣接 2 点間の経路の総数の上界の証明

定理 2.1. $N \times N$ 格子グラフにおける隣接 *2* 点間を結ぶ経路の総数の上界は $2^{N^2}+4$ である.

柴田らの色塗りの考え方を用いて, まずは以下二つの補題を証明する.

補題 2.2. $N \times N$ 格子グラフ中の単純無向閉路の総数の上界は 2^{N^2} である.

証明 . 上記の柴田らの色塗りの定理を単純無向閉路に用いて経路の総数の上界の証明を行う. 格子グラフの各セルに対してその内部に色付けを行う. 色付けされたセルが全て連結している時, そのセルの色付けされた部分とされていない部分の境界は一意な単純無向閉路を定める. よって, 格子グラフ内のセルの塗り方により単純無向閉路の形が定まる. 従って単純閉路の総数の上界は $N \times N$ のセルの色を塗るか塗らないかの 2 通りを考えれば良いため 2^{N^2} である. □

補題 2.3. 頂点 s の次数を k とする. 頂点 s の隣接点に *1* から k まで番号を振り分ける. 頂点 s を通過する単純無向閉路の総数は次の式を満たす.

$$|P_{cycle}| = |P_{s-1}| - 1 + |P_{s-2}| - 1 + ... + |P_{s-k}| - 1$$

証明 . 任意の閉路はちょうど *2* つの向きを持つので, 右回りとして考える. 有向閉路を作る場合, 経路の都合上, ある頂点 s に対する隣接点をその始点から出る場合と入る場合の *2* 点を通る必要があり, 頂点 s に戻るために最後に通る頂点を j とする *(j=1,2...k).s* から j への単純経路の集合 P_{s-j} には *s,j* という経路が含まれるため, これを除く必要がある. つまり, 頂点 s を通過し j から s に戻る有向閉路の総数は $|P_{s-j}|-1$ と等しい.

以上の議論から

$$|P_{cycle}| = |P_{s-1}| + |P_{s-2}| + \cdots + |P_{s-k}| - k$$

が導かれる. □

以上 2 つの補題より以下の定理 3.1.1 の証明を行う.

証明 (定理 2.1.1)**.** 補題 *2* の左辺は 2^{N^2} 以下である. さらに $|P_{cycle}|+k \geq |P_{s-k}|$ である. 格子グラフでは $k \leq 4$ のため隣接 *2* 点間の経路の総数の上界は $2^{N^2}+4$ である. □

3. 各頂点の訪問単純閉路数に対するワンパス ZDD アルゴリズム

ある 1 点を含む単純閉路の総数を各頂点に対して考える場合, 頂点数回 CyclePath アルゴリズムを動かす必要があった. 本稿では入力されたグラフ G に対して G における全通りの閉路を列挙すると同時に,G における各頂点 $v \in V$ が持つ閉路の数をワンパスで出力する CycleZDD を提案する.

3.1 アルゴリズムの概要

今回提案する手法である CycleZDD の擬似コードを Algorithm3.1 に表す.

アルゴリズムの入力値と出力値は以下である.

- 入力値:グラフ $G = (V,E)$
- 出力値:各単純閉路が通過する各頂点のリスト

格子グラフ内に存在する全ての単純閉路の集合を P と定義する. また, その要素 $c \in P$ は単純閉路を満たす頂点集合 $\{v_1, v_2 \ldots, v_k\}$ と辺集合 $\{e_1, e_2 \ldots, e_k\}$ の集合族である.

3.2 ある 1 本の単純閉路に対する処理

1 本の単純経路の処理に対して, その経路が通過する頂点

Algorithm 3.1 CycleZDD

Input: グラフ G
Output: 各頂点の保有する単純閉路数
キー = 頂点番号, 値　= 経路の総数となるハッシュ H を定義する
for each 辺 e ∈ E **do**
e の端点 = x,y
$H[x]$, $H[y]$ += 現在のノードが持つ経路数
for each ノード N ∈ P **do**
if 単純閉路 **then**
1 終点へ接続
else { サイクル不可 }
0 終点へ接続
else
頂点の連結成分と頂点のパス保有数を次ノードに付与
if 同一経路がある **then**
パスの数と共に頂点のパス保有数を足し合わせる
end if
end if
end for
end for

の情報を記録する方法を解説する. 格子グラフの各頂点に 1 から頂点数 N^2 までの番号を振り当てる. 次に, 各頂点が通過されたという情報を持つためにキーを頂点番号, 値を経路の総数となるハッシュを定義する. 辺 e∈E の端点の番号を x,y と置き, 辺 e を採用した場合,x,y をキーに持つ値のカウントをそれぞれ 1 増やす. よって,E の数ループを行い, ある単純閉路 p における辺集合 $\{e_1, e_2 \ldots, e_k\}$ を採用した場合, その p がもつ頂点集合 $\{v_1, v_2 \ldots, v_k\}$ の値全てに対してカウントがちょうど 2 ずつ増えているため,1 つの単純閉路においてその経路が通過する頂点は正確に数え上げられる.

3.3 ZDD への拡張

次に ZDD のデータ構造において経路が通過する頂点の情報を組み合わせることが可能であることについて説明する. 前提として,CyclePath アルゴリズムによって, 格子グラフ内の単純閉路の列挙のための ZDD の各ノードはフロンティア法に基づき合成される際, 正確に経路の本数を所持している.

辺を 1 本採用した場合, x,y をキーに持つ値をその地点におけるノードが保有する経路数増やす. ノードがマージされる場合, 同一キーに対しては値同士を加算し,1 方にキーが含まれない場合は片方の値のみをコピーする. 従って, フロンティア法により同一経路とみなされていながら, 実際は異なる経路に関しても経路の通過する頂点を正確にカウントできる.

4. 計算機実験と考察

4.1 計算時間の比較

フロンティア法と分岐条件の変更により CyclePath アルゴリズムを作ることが可能とであるが, ここでも全頂点数 $|V|$ の回数 ZDD を構築する必要がある.

今回提案する CycleZDD アルゴリズムにより各頂点を含む単純閉路数を列挙するためには ZDD を 1 度だけ構築すれば良いため, 単純閉路の列挙にかかる時間は頂点数に比例する高速化が見込める.

計算機により計測した時間の比較は以下のようである.

表 1: アルゴリズムによる計算時間

格子サイズ	Lemma2	ZDD	CycleZDD
N = 5(36)	20.180s	2.5921s	0.324608s
N = 6(49)	139.23s	28.186s	0.576267s
N = 7(64)	17.6min	109.56s	1.41316s
N = 8(81)	115min	497.55s	4.24655s
N = 9(100)	24h	2632.9s	18.5794s

4.2 ヒートマップ

また, CycleZDD を用いて出力された格子グラフにおける単純閉路数のヒートマップは以下のようなものである. 便宜上これまでの経路探索問題における頂点を格子 1 つに対応させており, 格子を通過する単純閉路の総数のヒートマップを作成している. 色の濃いほど単純閉路の総数が多くなっており, 最も外側の格子から 1 周目, 2 周目と定義した場合, 2 周目が最も単純閉路の総数が多く分布しており, 3 周目に位置する頂点を通過する単純閉路の総数が小さくなりまた中心に従って単純閉路の総数が大きくなる特徴が見られる格子に右下に記述してある数字はその格子を通過する単純閉路の総数である.

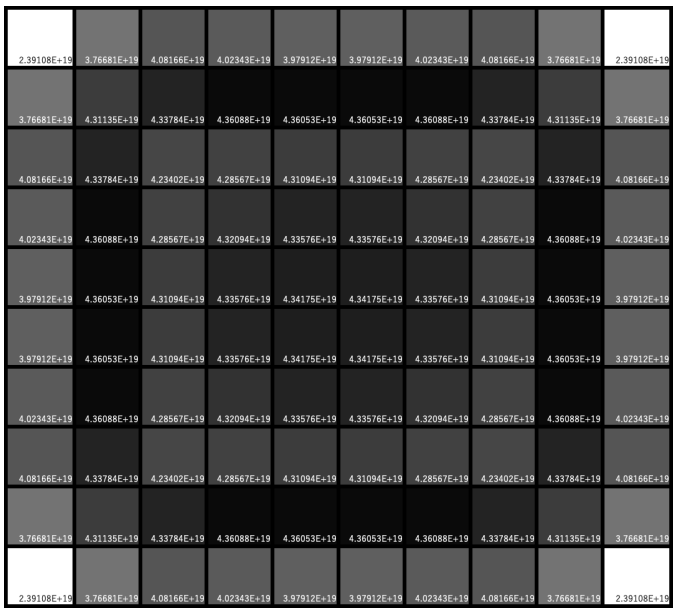

図 2: 10×10 頂点におけるヒートマップ

4.3 SimplePath との関係

s-t 間の SimplePath の経路の総数は次数の関係から, 内部 2 点間の方が経路の総数が多いと予想していた. しかし実際に SimPath アルゴリズムにより s-t が左上端と右下端に位置する時の経路の総数と s,t がそれぞれ斜め右下, 斜め左上に移動した場合の 2 点間の経路の総数は以下のようになっている. 格子グラフの左上端の頂点を (0, 0) 頂点, 右下端を (N,N) 頂点, (0, 0) 頂点の右斜下を (1, 1) 頂点, (N,N) 頂点の左斜上を ($N-1$,$N-1$) 頂点とする. 格子グラフの外周上の点を通る単純閉路の総数が少ないことに対して, 単純経路では外周上を結ぶ経路の総数がもっとも多い結果となった.

表 2: 各経路による経路の総数

格子サイズ	(0,0)-(N,N)	(1,1)-($N-1$,$N-1$)
N = 5(36)	1262816	592094
N = 6(49)	575780564	279466661
N = 7(64)	789360053252	514006687590
N = 8(81)	3266598486981642	1378292310954861
N = 9(100)	41044208702632496804	18910494162806248370

4.4 単純閉路が通過する頂点の考察

次に各頂点 (1,···,81) を通過する単純閉路の個数をプロットしたものが図 5.7 である.

上側の線が 1 点を通過する単純閉路の総数であり, 下側は格子内の単純閉路がもっとも多く通過する 1 点と各頂点 2 点を通過する単純閉路の総数である. 経路作成の性質上, 下線である, 選択された 1 点とその隣接点を通る経路の総数は大きくなっているものの, 上線の各頂点を通過する単純閉路の総数の分布と同様の形状を示している. 従って格子グラフ内のどの頂点に対しても, 同程度の割合で自身と他の頂点 2 点を通過する単純閉路を作成することができるとわかる.

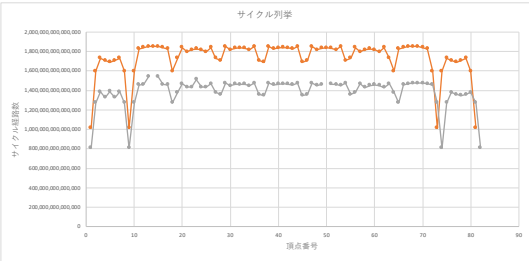

図 3: プロット図
Fig. 3 Plot diagram

5. おわりに

本研究は格子グラフにおける単純経路の総数は s,t の 2 点がグラフ内部に位置している場合の方が経路の総数が多いという予想のもと始まった. 計算機実験により最大の s,t 単純経路はお姉さん問題において考えられた s,t が左上端と右下端に位置する場合の経路であるという計算機実験結果を得た. 数学的証明が今後の課題である. 本論文の結果として, 格子グラフにおける単純閉路の総数の上界と, 隣接 2 点間の経路の総数の上界が 2^{N^2} であることを示した. また, 単純閉路の列挙において, 各頂点が保有する単純閉路をワンパスで求める手法を提案した. 今後の課題として今回提案したアルゴリズムにより出力されたヒートマップの特徴を解明することが挙げられる.

参考文献

[1] 川原 純, 斎藤 寿樹, 鈴木 拡, 湊 真一, 吉仲 亮,ZDD によるパスの列挙, 数理解析研究所講究録, 第 1744 巻, 35-41,2011
[2] D.E. Knuth, The Art of Computer Programming, 4-1,Addison-WesleyProfessional, 2009.
[3] S. Minato. Zero-Suppressed BDDs for Set Manipulation in Combinatorial Problems. In *DAC*, 272-277, 1993.
[4] 湊真一 (編)ERATO 湊離散構造処理系プロジェクト (著), 超高速グラフ列挙アルゴリズム: < フカシギの数え方 > が拓く, 組合せ問題への新アプローチ, 森北出版, 2015.
[5] 柴田友樹, 山内由紀子, 来嶋秀治, 山下雅史, 格子上の経路数え上げの乱択近似–24 時間で解くおねえさん問題, 2015 年度 冬の LA シンポジウム, 京都大学数理解析研究所, 2016 年 1 月 28 日.
[6] 高野圭司, フロンティア法から生成される ZDD の幅解析, 数理解析研究所講究録 第 1849 巻, 77-82,2013.
[7] Graphillion - 無数のグラフを効率的に扱うための高速・軽量ライブラリ https://github.com/takemaru/graphillion/wiki
[8] 『フカシギの数え方』 おねえさんといっしょ! みんなで数えてみよう! (https://www.youtube.com/watch?v=Q4gTV4r0zRs)