Vol.2014-AL-147 No.11
2014/3/4

キャタピラグラフの独立点集合遷移問題に対する多項式時間アルゴリズム

山田　武[†1]　　上原隆平[†2]

概要：グラフGに対し，｜II_b｜＝｜II_r｜であるような２つの独立点集合II_bとII_rが与えられたとする．また，Gにおいて，II_bに含まれる各点にはトークンが置かれているとする．スライディングトークン問題とは，II_bからII_rへのGの独立点集合の系列が存在するか判定する問題である．ただし，系列に含まれる各独立点集合は，その１つ前の独立点集合から，ただ１つのトークンをGの辺に沿ってスライドさせることで得られなければならない．本論文では，キャタピラグラフに対して，この問題が線形時間で解けることを示す．

キーワード　　グラフアルゴリズム，スライディングトークン，遷移問題，独立点集合

Linear time algorithm for Independent Set Reconfiguration Problem on a caterpillar graph

Takeshi　YAMADA[†1],　Ryuhei　UEHARA[†2]

Abstract　Suppose that we are given two independent sets II_b and II_r of a graph such that ｜II_b｜＝｜II_r｜, and imagine that a token is placed on each vertex in II_b. Then, *SLIDING TOKEN* is to determine whether there exists a sequence of independent sets which transforms II_b into II_r so that each independent set in the sequence results from the previous one by sliding exactly one token along an edge in the graph. In this paper, we show that the problem is solvable in linear time for caterpillar graphs.

Key words　graph algorithm, independent set, reconfiguration problem, sliding token

1. はじめに

スライディングトークン問題は Hearn と Demaine により考え出されたものである[1]．グラフの独立点集合とは，集合に属する，どの２つの頂点も互いに隣接していない状態である．｜II_b｜＝｜II_r｜である，２つの独立点集合II_bとII_rが与えられた時，スライディングトークン問題とは，独立点集合の列＜$\mathrm{II}_1,\mathrm{II}_2,\ldots,\mathrm{II}_\ell$＞が存在するかを決定する問題である．ただし，ここで，以下（a），（b）が成立しているものとする．

（a）$\mathrm{II}_1=\mathrm{II}_b$，$\mathrm{II}_\ell=\mathrm{II}_r$，$1\le i\le \ell$のすべての$i$について
｜$\mathrm{II}i$｜＝｜II_b｜＝｜II_r｜

（b）$2\le i\le \ell$のそれぞれのiについて，$\mathrm{II}i$はII_{i-1}から次の操作により得られる．II_{i-1}に属する頂点u上のただ１つのトークンをその隣接する頂点vに，辺$\{u,v\}$に沿ってスライドさせる．

すでに，proper interval graph と trivially perfect graph については，この問題が$O(n)$時間で解けることが示されている[2]．

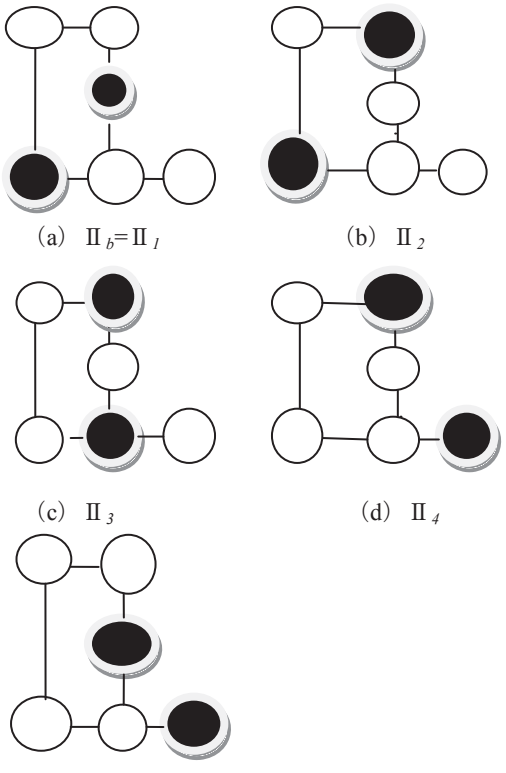

図１　同一グラフの独立点集合の列（●：独立点集合の頂点）

†1 北陸先端科学技術大学院大学　情報科学研究科
School of Information Science，JAIST
E-mail: tyama@jaist.ac.jp

†2 北陸先端科学技術大学院大学　情報科学研究科
School of Information Science，JAIST
E-mail: uehara@jaist.ac.jp

2. キャタピラグラフのスライディングトークン問題

2.1 キャタピラグラフ

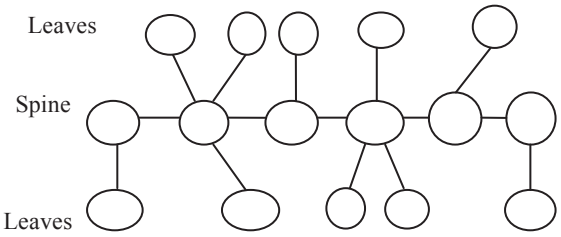

図 2　キャタピラグラフの一例

キャタピラグラフとは，path に次数 1 の頂点がつながった木である．正確には，グラフ G=(V, E)の頂点集合 V は Spine S とそれ以外の L に分別でき，S は path(s[1]，…，s[m])を導出し，L の要素は次数 1 で S の要素につながっている．Spine の頂点数，Spine の両端の頂点の次数は各々２以上とする．(図 2 参照)

2.2 スライディングトークン問題

定理１

キャタピラグラフ G と２つの独立点集合 II_b と II_r に対するスライディングトークン問題は，$O(n)$時間及び $O(n)$領域で解くことができる．ここで n はキャタピラグラフの頂点の数である．さらに，遷移が可能である場合，再配置の列は $O(n^2)$時間及び O(n)領域で出力可能である．

補題１

キャタピラグラフ G と２つの独立点集合 II_b と II_r に対するスライディングトークン問題は，別のキャタピラグラフ G’と２つの独立点集合 II'_b, II'_r に線形時間で還元することができる．ただし，ここで下記(1)～(3)が成立する．

(1)G，II_b と II_r に対するスライディングトークン問題が遷移可能である⇔G’，II'_b と II'_r に対するスライディングトークン問題が遷移可能である．

(2)G’の頂点の最大次数は高々3 である．

(3) 還元後のキャタピラグラフ G’の Spine の両端の頂点の次数は各々２である．

言い換えれば，キャタピラグラフに対するスライディングトークン問題は，最大次数 3 のキャタピラグラフに限定して考えれば十分である．

（補題 1 の証明）

G 上で s[i]を 4 以上の次数を持った頂点とする．そのとき，s[i]には少なくとも，2 つの葉ℓ[i]，ℓ’[i]がつながっている状態である．今，II_b に関する２つのトークンがℓ[i]とℓ’[i]にあるとする．すると，この２つのトークンは動かせない．また，他のトークンもこの２つのトークンによってブロックされているので，s[i]を通過することは出来ない．II_r もまた，この２つのトークンを含むならば，s[i]とその葉を G から取り除き，新たな２つのキャタピラグラフの独立点集合遷移問題として取り扱うことができる．II_r がこの２つのトークンを含まなければ，問題に対する答えが‘遷移不可能’であることは明らかである．少なくとも２つのトークンが，II_b 側のキャタピラグラフの１つの S の頂点の葉として共存するのであれば，それを線形時間で識別できる．そこで，II_b，II_r とも，１つの頂点にトークンを持った葉は高々１本しかつながっていないとすることができる．同様の理由により，１つの頂点につながっている１本の葉以外はこの問題を考える時，除外してよい．より正確には，II_b が II_r に帰着可能であるにせよ，そうでないにせよ，１つの S の頂点につながっている葉のうち，使用されるのは１本であるということである．それゆえ，１つの頂点から，上記１本の葉以外は除去できる．特に，不要な葉を取り除き，s[1] 及び s[m]の次数を 2 とすることできる．　□

以上より，これからは，補題１で述べられたキャタピラグラフのみを考えていくことにする．つまり，Spine の頂点 s[1]，s[2]，…，s[m]を持ったキャタピラグラフに対して，s[1] 及び s[m]の次数は 2，1＜i＜m である i に対しては s[i]の次数は 2 ないし 3 とする．また，Spine の頂点 s[i]に葉がある場合，ℓ[i]とする．

ここで，このキャタピラグラフの独立点集合遷移問題にとって核となる概念を定義する．

キャタピラグラフ G と G の独立点集合Ⅱに対して，G 上の 1 本のパス P＝(p[1]，p[2]，…，p[k])がⅡによって lock されている(locked path)とは下記の３つの条件が同時に成立することである．

(a) k は 2 以上の奇数である．

(b) Ⅱ∩P＝(p[1]，p[3]，p[5]，…，p[k])

(c) p[1]と p[k]の次数は 1 である．（つまり，両者とも葉である．）

locked path の概念を用いながら，キャタピラグラフ上の動かすことのできない独立点集合というものを特徴づける．

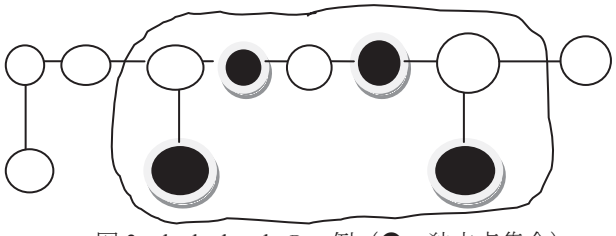

図 3　locked path の一例（●：独立点集合）

定理 2
キャタピラグラフを G，G の独立点集合をⅡとする．このとき，Ⅱに属するトークンをまったく動かすことができない⇔ある h に対して，locked path の集合 P[1]，…，P[h]が存在し，独立点集合Ⅱはその和集合である．

（定理 2 の証明）
明らかに，Ⅱが locked path の和集合であるならば，トークンを動かすことはできない．それゆえ，G 上のあるⅡに対して，Ⅱに属するどのトークンも動かすことができないということを仮定し，それらはすべて，locked path 上に置かれているということを示す．
まず最初に，トークンが葉$\ell[i]$上にある場合を考える．そのとき，s[i]にトークンは存在しない．s[i-1] ，s[i+1]の両方にトークンが存在しないならば，s[i]にトークンを動かすことができるが，これは矛盾である．ここで，一般性を失うことなく，s[i+1]にトークンが存在すると仮定できる．その結果，$\ell[i+1]$, s[i+2]にトークンが存在しないことになる．$\ell[i+2]$にトークンが存在するならば，トークンは locked path($\ell[i]$, s[i], s[i+1] , s[i+2], $\ell[i+2]$)上に置かれているとなり，矛盾は生じない．
次に，$\ell[i+2]$にトークンが存在しないと仮定する．その時，s[i+1]上のトークンは動かすことができないので，s[i+3]上にトークンが存在しなければならない．このプロセスを繰り返すことにより，結果的に，1 つの端点$\ell[i]$を持つ，locked path を得る．次に，トークンが Spine の頂点 s[i]に存在する場合を考える．s[i]が葉$\ell[i]$を持つならば，トークンを$\ell[i]$に動かすことができる．これは矛盾である．よって，s[i]は葉を持たない．一方，s[i-1]，s[i+1]の両方にトークンは存在しない．これら，2 つの頂点に関して，最初の場合と同様の手法が取れるので，(s[i-1], s[i], s[i+1])を path の一部とする，locked path を得ることができる．以上で証明は完了である． □

キャタピラグラフ G と独立点集合Ⅱについて，Ⅱが locked path を含むならば，その locked path の頂点を通過して，トークンを動かすことはできない．ゆえに，locked path は G を 2 つの部分グラフに分割すると考えてよい．このとき，それぞれの独立点集合遷移問題を考えればよいことになる．

キャタピラグラフ G と正の整数 k について以下，サイズが k である，最左充填独立点集合Π_k(G)を定義する．
まず最初に，Π_k(G)に$\ell[1]$を加える．そして，各々の i=2, 3, 4, …, m-1 について，k 個のトークンを次のルールに従って置いていく．
(1) s[i-1]∉$\Pi_k(G)$ ＆ s[i]の次数は 2
　s[i]を$\Pi_k(G)$に加える．
(2) s[i-1]∉$\Pi_k(G)$ ＆ s[i]の次数は 3
　$\ell[i]$を$\Pi_k(G)$に加える．
(3) s[i-1]∈$\Pi_k(G)$
　この i に関しては何もしない．

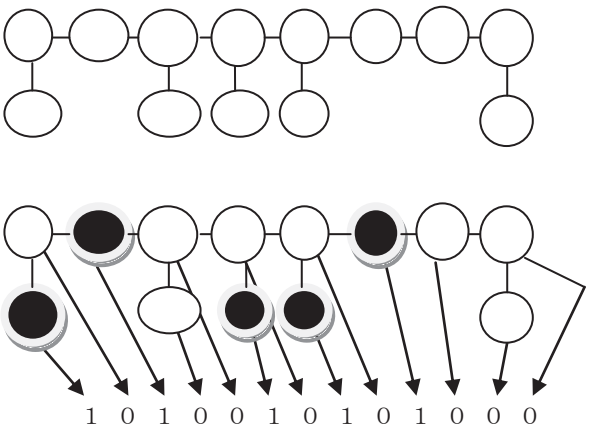

図 4 キャタピラグラフ，サイズ 5 の最左充填独立点集合及びその 2 進符号化

定理 3
キャタピラグラフ G とサイズが k の独立点集合Ⅱに対して，Ⅱが locked path を含まないとする．そのとき，Ⅱは最左充填独立点集合$\Pi_k(G)$に遷移可能である．さらに，遷移の回数は $O(n^2)$である．ここで，n はキャタピラグラフ G の頂点の数である．
(定理 3 の証明)
まず最初に，G について，以下のような全順序を定義する．
$\ell[1]$< s[1]<$\ell[2]$< s[2]< … < $\ell[i]$< s[i]< … <ℓ[m]< s[m]
(ただし，$\ell[i]$に関しては，存在しない場合がある．)
ここで，2 進数を表す列，B(Ⅱ)＝b[0] b[1] …b[n]を以下のルールにより定義する．(n はキャタピラグラフ G の頂点の数である．)
(0) 各々のビット b[i]は全順序で i 番目に現れる頂点に対応する．
(1) トークンを持った頂点は 1 に符号化される．
(2) トークンを持たない頂点は 0 に符号化される．

一例として，図 4 のキャタピラグラフのサイズ 5 の最左充填独立点集合は 2 進数列 10100101000 に符号化される．
k 個の頂点からなる最左充填独立点集合が形成する 2 進数列 B($\Pi_k(G)$)と，他の k 個の頂点からなる独立点集合が形成する 2 進数列 B(Ⅱ)について，各々を 2 進数とみなせば，必ず，B($\Pi_k(G)$)> B(Ⅱ)が成立する．1 を同数 k 個含む 2 つの 2 進数列 B_1 と B_2(｜B_1｜＝｜B_2｜とする．)について，以下のルールで決定される距離 dist (B_1, B_2)を導入する．
各々の i=1, 2, …, k について，B_1 が i 番目の 1 を $j_1(i)$番目のビットに含み，B_2 が i 番目の 1 を $j_2(i)$番目のビットに含むと仮定する．そのとき，dist (B_1, B_2)を
$\Sigma_{(i=1\cdots k)}\,\delta(j_1(i)-j_2(i))$

によって定義する．(δ は $\delta(0)=0$， $\delta(i)=1(i\neq 0)$によって定義される指示関数である．)
例えば，dist(10100, 00101)=2 となり，
dist (B_1, B_2)=0 ⇔ $B_1=B_2$ である．
さらに，2 つの長さ n の 2 進数列 B_1 と B_2 について，各々のビットは上記で導入した距離に対して高々1 しか寄与しないので，dist (B_1, B_2)は $O(n)$である．
次に，locked path を含まないサイズ k の独立点集合Ⅱと最左充填独立点集合$\Pi_k(G)$を比較する．そのとき，$\Pi \neq \Pi_k(G)$であり，2 進数 B($\Pi_k(G)$)>B(Ⅱ)である．
B($\Pi_k(G)$)= b[1] b[2] …b[n]，B(Ⅱ)= b'[1] b'[2] …b'[n]で表す．このとき, b[i]=1, b'[i]=0, すべての $i'(<i)$ について b[i']= b'[i']であるようなインデックス i が存在する．そのインデックス i について，B(Ⅱ)と B($\Pi_k(G)$)は同数の 1 を含むので，すべての $i\leq i'<j$ について b'[i']=0 及び b'[j]=1 であるような、もう 1 つのインデックス $j>i$ が存在する．
そこで，以下、2 つの場合(a), (b)が生じる．
(a)まず最初に，b'[j]が Spine の頂点 s[j']に対応する場合である．そのとき，s[j']と b[i]に対応する頂点の間の頂点にトークンは存在しない．一方，b[i]は Spine の頂点 s[i']か葉ℓ[i']に対応する．(ここで，i'は $i'<j'$を満たす)
どちらにせよ，$\Pi_k(G)$は最左充填独立点集合であるから，s[i-1]にトークンは存在しない．それゆえ，Ⅱ上で s[j']上のトークンを Spine の頂点 s[i']か葉ℓ[i']にスライドさせることができる．$O(n)$回のスライド作業の後に，新たな独立点集合Ⅱ'を得る．ここで，dist(Ⅱ', $\Pi_k(G)$)= dist(Ⅱ, $\Pi_k(G)$)−1 である．
(b)次に，b'[j]が葉ℓ[j']に対応する場合を考える．Ⅱは独立点集合であるから，s[j']にトークンは存在しない．さらに s[j'+1]にもトークンが存在しないならば，ℓ[j']上のトークンを s[i]に最初の場合と同様の手法でスライドさせることができる．そして，新たな独立点集合Ⅱ'を得る．
ここで，dist(Ⅱ', $\Pi_k(G)$)= dist(Ⅱ, $\Pi_k(G)$)−1 である．
そこで次に，s[j'+1]にトークンが存在する場合を考える．仮定により，Ⅱは locked path を含まない．よって，{ℓ[j'], s[j'], s[j'+1], s[j'+2]} を含むどんな path も locked path を含まない. それゆえ, s[j'']にトークンが存在し, {s[j''+1], ℓ[j''+1], s[j''+2]}にはトークンが存在しないようなインデックス $j''(j''>j')$が必ず存在する. そうすると, その path P に沿って，トークンを 1 つ 1 つ，ずらすことができ，$O(n)$回のスライド作業により, s[j'+1]上のトークンを s[j'+2]へ動かすことができる．そこで，ℓ[j']上のトークンを s[i]に最初の場合と同様の手法でスライドさせることができる．この作業の後，path P に沿って，トークンの遷移を巻き戻す．$O(n)$回のこの巻き戻し作業の後，新たな独立点集合Ⅱ'を得る．
ここで，dist(Ⅱ', $\Pi_k(G)$)= dist(Ⅱ, $\Pi_k(G)$)−1 である．

以上の過程を繰り返すことにより，最終的に，$O(n^2)$回のスライド作業の後，$\Pi_k(G)$を得る． □

最後に本論文の核となる定理 1 の証明に戻る．
（定理 1 の証明）
キャタピラグラフ上に与えられた独立点集合Π_bについて，各々の頂点が locked path の一部であるかどうか，$O(n)$時間で判定できる．
(0) 状態 S を“not locked path”で初期化する．
(1) $i=1, 2, \cdots, m$ について，(s[i], ℓ[i])の状態により状態 S を更新する．
(s[i], ℓ[i])=(x,y)は次のように決定される．
$x\in\{0,1\}$，$y\in\{0,1,-\}$
1 はトークンが頂点に存在することを示し，0 はトークンが頂点に存在しないことを示し，－は葉が存在しないことを示す．
(s[i], ℓ[i])が各々の場合について，状態 S を以下のように更新する．

Case(0,1):
S が“not locked path”であるならば，S を“locked path？”にセットする．そして，i を locked path の左端点候補として記憶する．S が“locked path？”である場合，すぐ隣の左の点にトークンが存在する場合，locked path が成立するので，左端点候補として記憶した点からこの点までが locked path である．すぐ隣の左の点にトークンが存在しない場合，いったん，S は“locked path？”をリセットし，この点が左端点候補となり，再び S は“locked path？”の状態となり，右側に進んで行く．

Case(0,0)及び Case(0,－):
s[i-1]にトークンが存在する場合，S の状態をそのまま，次に送る．s[i-1]にトークンが存在しない場合，
“not locked path”により，S をリセットする．
(S の前の状態がどのような状態であってもリセット.）

Case(1,0):
“not locked path”により，S をリセットする．
(S の前の状態がどのような状態であってもリセット.）

Case(1,－):
S の状態をそのまま次に送る．

上記の方法により，$O(n)$時間ですべての locked path の頂点を調べることができる．
まず最初に，この方法を，(G,Π_b)と(G,Π_r)に対して同時に適用する（要する時間は $O(n)$）．そして，両者の locked path が完全一致するかどうか調べる．完全一致しなければ，アルゴリズムは“遷移不可能”の答えを返す．

両者の locked path が完全一致の場合は，アルゴリズムはキャタピラグラフ G をいくつかの部分グラフに分割する(この部分グラフ群は locked path ではない頂点のみから形成される)．そして，各部分グラフごとに，遷移問題を解き進める．ここで，locked path のトークンが存在する左端点と右端点は必ず葉である．つまり，locked path を P=(p[1], p[2], …, p[k])とし，G が P によって G_1 と G_2 に分割されるのであれば，G_1 と G_2 の P に対する隣接点は p[2]と p[k-1]である．そして，この 2 点にはトークンは存在しない．よって，G_1 と G_2 は P の部分は気にせずに完全に分離して，問題を解けばよい．

次に，各々の部分グラフについて，アルゴリズムは II_b と II_r が同数のトークンを有するかを調べる．すべての部分グラフについて，トークンの数が一致すれば，アルゴリズムは“遷移可能”の答えを返し，そうでなければ，“遷移不可能”の答えを返す．上記アルゴリズムの正当性は定理 2 と定理 3 よりすぐに導かれる．また，アルゴリズムが $O(n)$時間，$O(n)$領域で実行されるのも容易にわかる．また，遷移列自体を出力するために，定理 3 を用いアルゴリズムに修正を加えれば，修正後のアルゴリズムは長さ $O(n^2)$の列を出力することになる． □

3. おわりに

キャタピラグラフの独立点集合遷移問題に関して，II_b から II_r へ遷移可能である場合，最短の系列を見つける手法を確立することが今後の課題である．また，木構造上の問題が多項式時間で解けるかどうかも今後の課題である．

参考文献

[1] Hearn, R.A., Demaine, E.D.: PSPACE-completeness of sliding-block puzzles and other problems through the nondeterministic constraint logic model of computation. Theoretical Computer Science 343, pp. 72-96(2005)

[2] Erik D.Demaine, Martin L.Demaine, Takehiro ITO, Hirotaka ONO,and Ryuhei UEHARA: Algorithms for Independent Set Reconfiguration Problem on Graphs
信学技報 Vol.113 No.371 電子情報通信学会技術研究報告
(COMP2013-38－COMP2013-59) COMP2013-39 pp.7-14