漏えい鍵共有直並列グラフからの鍵生成について

佐々木 達也[1]　林 優一[2]　水木 敬明[3]　曽根 秀昭[3]

概要：n 人のプレイヤーのうち，いくつかのペアが事前に 1 ビットの鍵を共有しており，それらの鍵が独立にある確率で盗聴者 Eve に漏えいすると仮定する．このような状況を漏えい鍵共有グラフで表す．このとき，ある二人のプレイヤーが他のプレイヤーと協力し，Eve への漏えい確率ができるだけ小さい 1 ビットの鍵を生成する問題を考える．この問題を解決するプロトコルとして st-フロープロトコルが存在するが，任意のグラフについてその漏えい確率を具体的に計算するのは困難である．本稿では，漏えい鍵共有直並列グラフに対象を絞り，生成する鍵の漏えい確率を効率的に求める手法を提案する．

キーワード：秘匿増幅，秘密鍵共有，鍵共有グラフ

1. はじめに

二人以上のプレイヤーからなるグループと盗聴者 Eve を仮定する．グループ内のいくつかのペアは事前に 1 ビットの秘密鍵（以下，**事前鍵**と呼ぶ）を共有しているものとする．このとき，各プレイヤーを点 $v \in V$ とし，事前鍵を共有しているプレイヤー同士を辺 $e \in E$ で結ぶことで，プレイヤーと事前鍵の共有関係を無向グラフ $G = (V, E)$ で表すことができる．このようなグラフ G を**鍵共有グラフ**と呼び，各辺 $e \in E$ に対応する事前鍵を $k_e \in \{0, 1\}$ で表す．例として図 1 に示すような鍵共有グラフ G^{ex} を考えよう．グラフ G^{ex} は，s，v，t という三人のプレイヤーにおいて，各ペアが事前鍵 k_{st}，k_{sv}，k_{vt} をそれぞれ共有していることを示している．

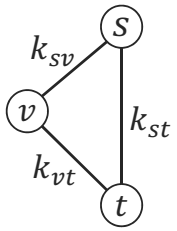

図 1　鍵共有グラフ $G^{\mathrm{ex}} = (\{s, t, v\}, \{st, sv, vt\})$

事前鍵の共有は様々な方法（例えば，Diffie-Hellman 鍵交換，RSA 暗号，量子暗号，郵便，電子メールなど）で行われており，これらの事前鍵はある確率分布に従って Eve に漏えいすると仮定する（cf. [6], [7]）．すなわち，漏えい辺集合 $F \subseteq E$ がある**漏えい分布** $\mathcal{L}$ に従って生起し，その F に含まれる全ての事前鍵の値は Eve に知られているとする（$E - F$ に含まれる鍵については，Eve は 1/2 を超える確率でその値を当てられない）．以後，鍵共有グラフ G と漏えい分布 $\mathcal{L}$ の対 $(G, \mathcal{L})$ を**漏えい鍵共有グラフ**と呼ぶ．

本稿では，$(G, \mathcal{L})$ と二人のプレイヤー s，t が与えられたとき，この二人のプレイヤーが他のプレイヤーと協力し，より漏えい確率の低い秘密鍵（以後，**生成鍵**と呼ぶ）を新たに生成する問題を考える．Eve への漏えい確率を最小にする鍵生成プロトコルとして st-フロープロトコル [5] が知られており，その確率を $\mathcal{E}_{\text{st-flow}}(G, \mathcal{L}, s, t)$ と書き，文脈から明らかなときは $\mathcal{E}_{\text{st-flow}}(G, \mathcal{L})$ と書く．しかしながら，任意の $(G, \mathcal{L})$ に対して，その漏えい確率 $\mathcal{E}_{\text{st-flow}}(G, \mathcal{L})$ を具体的に計算する手法は知られていない．最近の研究で，漏えい分布を一様で独立と仮定し，鍵共有グラフを完全グラフ [13] や完全二部グラフ [12] に限定した場合に，漏えい確率を多項式時間で求める手法が提案された．鍵共有グラフが常に完全グラフや完全二部グラフになるとは限らないため，他のクラスの鍵共有グラフについても同様に漏えい確率を求めたいが，未解決である．

本稿では，まず上述の問題に対する計算複雑性について考察する．具体的には，漏えい鍵共有グラフ $(G, \mathcal{L})$ の $\mathcal{L}$ が独立分布であると限定しても，st-フロープロトコルによる u の漏えい確率 $\mathcal{E}_{\text{st-flow}}(G, \mathcal{L})$ を求める問題は#P 完全であることを示す．これは 2 端子間ネットワーク信頼度 [2] を求める問題を帰着させることにより得られる．

また，完全グラフや完全二部グラフ以外のクラスとして

[1] 東北大学大学院　情報科学研究科
[2] 奈良先端科学技術大学院大学　情報科学研究科
[3] 東北大学　サイバーサイエンスセンター

直並列グラフを対象とし，直並列グラフの 2 端子間ネットワーク信頼度を線形時間で求めるアルゴリズム [9] が既に提案されていることを利用して，生成鍵 u の Eve への漏えい確率が効率的に求められることを確認する．

本稿における設定では，プレイヤーは認証付き公衆通信路を用いて通信する．すなわち，全てのメッセージは盗聴者 Eve を含めて全てのプレイヤーに同報されることに注意されたい．同じ状況を仮定する研究として秘匿増幅（Privacy Amplification）に関する研究がある（e.g. [1], [3], [11]）．

本稿の構成は次の通りである．まず，2 節では st-フロープロトコルの説明を行い，扱う問題を説明する．3 節では $\mathcal{E}_{\text{st-flow}}(G,\mathcal{L})$ を求める問題の計算複雑性について考察する．4 節ではグラフを直並列グラフに限定し，u の Eve への漏えい確率を効率的に求める手法を説明する．5 節では，いくつかの直並列グラフに対して提案手法を適用し，漏えい確率を与える多項式を具体的に示す．

2. st-フロープロトコル

st-フロープロトコル [5] は，全ての事前鍵を使用することで最も漏えい確率の低い生成鍵を共有できるプロトコルである．すなわち，st-フロープロトコルは任意の漏えい鍵共有グラフ $(G,\mathcal{L})$ において，どのようなプロトコル $\mathcal{P}$ に対しても

$$\mathcal{E}_{\text{st-flow}}(G,\mathcal{L}) \leq \mathcal{E}_{\mathcal{P}}(G,\mathcal{L})$$

である．

具体的に，図 1 に示した鍵共有グラフ G^{ex} に対して st-フロープロトコルを実行する例を示す．st-フロープロトコルは，次のようにプレイヤー s と t が v と協力して生成鍵 u を得る．

(1) プレイヤー s が 1 ビット u をランダムに選ぶ．
(2) u_1 をランダムに選び，$u_2 = u \oplus u_1$ とする．
(3) u_1，u_2 をプレイヤー v，t にそれぞれワンタイムパッド（OTP）暗号 [10] を用いて送信する．
(4) u_1 を受け取った v は，u_1 を t に OTP 暗号を用いて送信する．
(5) u_1，u_2 を受け取った t は，$u_1 \oplus u_2 = u$ を計算することで u を復元し，これを生成鍵とする．

このプロトコルを使用した場合，u_1，u_2 の両方が漏えいした場合にのみ，生成鍵 u は Eve に漏えいすることに注意しよう．

例えば，各事前鍵の漏えいは独立に発生すると仮定しよう．各辺 e に対応する漏えい確率を p_e と表すとき，鍵共有グラフ G^{ex} における漏えい分布は表 1 のようになる．このとき，u_1 が Eve に漏えいする確率は k_{sv}，k_{vt} のうち，少なくとも片方が漏えいする確率に等しいので，

$$1-(1-p_{sv})(1-p_{vt}) = p_{sv}+p_{vt}-p_{sv}p_{vt}$$

となる．u_2 が Eve に漏えいする確率は p_{st} なので，u が Eve に漏えいする確率は，

$$\mathcal{E}_{\text{st-flow}}(G^{\text{ex}},p_{st},p_{sv},p_{vt}) = p_{st}p_{sv}+p_{st}p_{vt}-p_{st}p_{sv}p_{vt}$$

となる．ただし，$(G,p_1,p_2,...,p_{|E|})$ は，各事前鍵 k_i が独立に確率 p_i で漏えいする漏えい鍵共有グラフを表すことにしている．

表 1 $(G^{\text{ex}},p_{st},p_{sv},p_{vt})$ の漏えい分布

漏えい辺集合	発生確率
$\{st,sv,vt\}$	$p_{st}p_{sv}p_{vt}$
$\{st,sv\}$	$p_{st}p_{sv}(1-p_{vt})$
$\{st,vt\}$	$p_{st}(1-p_{sv})p_{vt}$
$\{sv,vt\}$	$(1-p_{st})p_{sv}p_{vt}$
$\{st\}$	$p_{st}(1-p_{sv})(1-p_{vt})$
$\{sv\}$	$(1-p_{st})p_{sv}(1-p_{vt})$
$\{vt\}$	$(1-p_{st})(1-p_{sv})p_{vt}$
$\emptyset$	$(1-p_{st})(1-p_{sv})(1-p_{vt})$

G^{ex} のように小さいグラフの場合は，このようにして $\mathcal{E}_{\text{st-flow}}(G^{\text{ex}},p_{st},p_{sv},p_{vt})$ を求められるが，任意のグラフ G に対しては，次のような特徴付けが存在する．すなわち，生成鍵 u が Eve に漏えいするかどうかについては，漏えい辺集合のグラフ理論的性質によって特徴付けられる [5]．具体的には図 2(a) のように漏えい辺集合 F が s と t を分離するとき（カットであるとき），u は Eve に漏えいする．また，図 2(b) のように漏えい辺集合 F が s と t を分離しないとき，u は Eve に漏えいしない．言い換えると，$G=(V,E)$ と漏えい辺集合 F が与えられたとき，E から F を除いて得られるグラフ $(V,E-F)$ において，s と t が非連結であるとき u は Eve に漏えいし，連結であるとき漏えいしない．

いま，漏えい鍵共有グラフ $(G,\mathcal{L})$ に対し，s と t を分離する漏えい辺集合の集合を

$$\text{Sep}(s,t;G) = \{F \subseteq E \mid F \text{ は } s \text{ と } t \text{ を分離する }\}$$

と定義し，$\Pr(F)$ を漏えい辺集合 F の生起確率とする．このとき，st-フロープロトコルを実行した場合の，生成鍵 u が Eve に漏えいする確率は

$$\mathcal{E}_{\text{st-flow}}(G,\mathcal{L}) = \sum_{F\subseteq \text{Sep}(s,t;G)} \Pr(F)$$

で求められる．すなわち，s と t を分離する漏えい辺集合の生起確率の総和が u の Eve への漏えい確率になる．

次節では，各事前鍵が独立に漏えいすると仮定したとき，漏えい鍵共有グラフ $(G,p_1,...,p_{|E|})$ に対して $\mathcal{E}_{\text{st-flow}}(G,p_1,...,p_{|E|})$ を求める問題が#P 完全であることを示す．

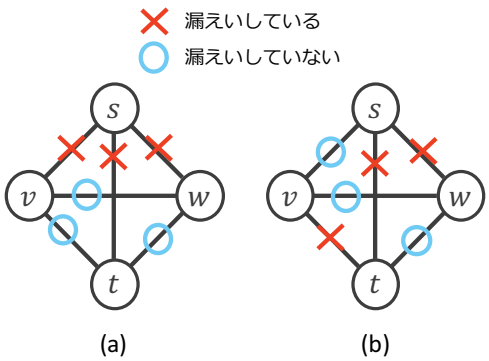

図 2　s と t が分離される場合とされない場合

3. 計算複雑性について

本節では，$\mathcal{E}_{\text{st-flow}}(G, p_1, ..., p_{|E|})$ を求める問題の計算複雑性について考察する．

まず，2 端子間ネットワーク信頼度問題について説明を行うために，以下のようなネットワークモデルを導入する．コンピュータを点 $v \in V$ で表し，接続しているコンピュータ同士を辺 $e \in E$ で結ぶことで，コンピュータネットワークを無向グラフ $G = (V, E)$ でモデル化する．このとき，各辺が独立にある確率で故障する（故障が発生した辺は取り除かれる）状況を考え，これを**確率グラフ** $(G, p_1, ..., p_{|E|})$ で表す．2 端子間ネットワーク信頼度とは，確率グラフ $(G, p_1, ..., p_{|E|})$ において，ある 2 端子間に道が存在する確率である．これは，2 端子間が連結であるような故障辺集合の生起確率の総和をとることで求められる．

具体例として，漏えい鍵共有グラフ $(G^{\text{ex}}, p_{st}, p_{sv}, p_{vt})$ を前述した確率グラフ $(G^{\text{ex}}, p_{st}, p_{sv}, p_{vt})$ とみなし，s-t 間ネットワーク信頼度を考えよう．当然ながら，$(G^{\text{ex}}, p_{st}, p_{sv}, p_{vt})$ における各故障辺集合とその発生確率は表 1 と同じになる．このとき，s-t 間が連結になる故障辺集合 $2^E - \text{Sep}(s, t; G^{\text{ex}})$ とその発生確率は表 2 のようになる．確率グラフ $(G^{\text{ex}}, p_{st}, p_{sv}, p_{vt})$ における s-t 間ネットワーク信頼度を $R(G^{\text{ex}}, p_{st}, p_{sv}, p_{vt})$ とすると，

$$\begin{aligned} R(G^{\text{ex}}, p_{st}, p_{sv}, p_{vt}) &= 1 - p_{st}p_{sv} - p_{st}p_{vt} + p_{st}p_{sv}p_{vt} \\ &= 1 - \mathcal{E}_{\text{st-flow}}(G^{\text{ex}}, p_{st}, p_{sv}, p_{vt}) \end{aligned}$$

と書ける．

表 2　s-t が連結になる故障辺集合

故障辺集合	発生確率
$\{sv, vt\}$	$(1-p_{st})p_{sv}p_{vt}$
$\{st\}$	$p_{st}(1-p_{sv})(1-p_{vt})$
$\{sv\}$	$(1-p_{st})p_{sv}(1-p_{vt})$
$\{vt\}$	$(1-p_{st})(1-p_{sv})p_{vt}$
$\emptyset$	$(1-p_{st})(1-p_{sv})(1-p_{vt})$

確率グラフ $(G, p_1, ..., p_{|E|})$ において，漏えい辺集合は s-t が連結になる集合と，非連結になる集合に分かれる．すなわち，確率グラフ $(G, p_1, ..., p_{|E|})$ が与えられたとき，$\mathcal{E}_{\text{st-flow}}(G, p_1, ..., p_{|E|})$ と $R(G, p_1, ..., p_{|E|})$ には以下の関係が成り立つ．

$$\mathcal{E}_{\text{st-flow}}(G, p_1, ..., p_{|E|}) + R(G, p_1, ..., p_{|E|}) = 1 \qquad (1)$$

よって，$\mathcal{E}_{\text{st-flow}}(G, p_1, ..., p_{|E|})$ を求める問題の計算複雑性は，$R(G, p_1, ..., p_{|E|})$ を求める問題の計算複雑性と同じである．また，$R(G, p_1, ..., p_{|E|})$ を求める問題は#P 完全であることが証明されている [8] ため，$\mathcal{E}_{\text{st-flow}}(G, p_1, ..., p_{|E|})$ を求める問題も#P 完全である．

したがって，任意の $(G, p_1, ..., p_{|E|})$ に対して $\mathcal{E}_{\text{st-flow}}(G, p_1, ..., p_{|E|})$ を求める多項式時間アルゴリズムは存在しそうにない．ただし，グラフ G を直並列グラフに限定したとき，$R(G, p_1, ..., p_{|E|})$ を線形時間で求めるアルゴリズムが提案されている [9]．そこで次節では，これを用いて漏えい鍵共有直並列グラフにおける漏えい確率を求める．

4. 直並列グラフに対する漏えい確率

本節では，鍵共有グラフを鍵共有直並列グラフに限定した場合に，効率的に漏えい確率を求めるアルゴリズムを記述する．

4.1 直並列グラフとアルゴリズムの概要

直並列グラフとは，2 つの点とそれを結ぶ 1 本の辺が与えられたとき，以下のいずれかの操作を繰り返すことで得られるグラフである．

- ある 1 本の辺の間に点を挿入し，2 本の直列な辺に置き換える．
- ある 1 本の辺を 2 本の並列な辺に置き換える．

これにより，例えば図 3 のような直並列グラフ G^{ex2} が得られる．当然ながら，得られたグラフに先ほどと逆の操作（2 本の直列な辺を 1 本の辺に置き換える．2 本の並列な辺を 1 本の辺に置き換える．）を繰り返すと，ある 2 点とそれを結ぶ 1 本の辺になるまで戻すことができる．

3 節の最後で述べたように，G が直並列グラフであるとき，$R(G, p_1, ..., p_{|E|})$ を求める線形時間アルゴリズム [9] が存在するので，もちろん $\mathcal{E}_{\text{st-flow}}(G, p_1, ..., p_{|E|})$ も線形時間で求めることができる．しかしながら，既存研究の文献 [9] では，より一般化された k 端子直並列グラフにおいて，k 端子間信頼度を線形時間で求めるアルゴリズムとして示されているため，$k = 2$ の場合には不要な箇所が含まれている．そこで，既存アルゴリズム [9] を精査し，端子数が 2 の場合に真に必要な部分を取り出し，$k = 2$ に特化したアルゴリズムを記述する．

主要なアイデアは，グラフ内のある 2 点間の漏えい確率を保ったまま，より小さいグラフに縮退させることである

[2], [9]．ただし，縮退を行う際，注目している 2 点が端子（本稿における s と t が該当する）である場合には取り除くことができないという制約があるので考慮が必要である．

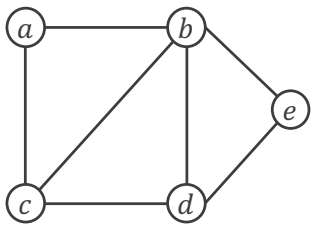

図 3　直並列グラフ G^{ex2}

4.2　3 つの縮退アルゴリズム

次に示す縮退アルゴリズム（と 4.3 節の縮退アルゴリズム）により $\mathcal{E}_{\text{st-flow}}(G, p_1, ..., p_{|E|})$ を再帰的に求めることができる．

- 次数-1 縮退
  図 4(a) のように $\deg(u) = 1$ の端子ではない点 u が存在するとき，u-v 間で通信は行われないため，グラフから点 u と辺 e_{uv} を除去する．また，そのような点 u が存在せず，図 4(b) のように次数 1 の端子 s（または端子 t）が存在する状況を考えよう．このときは，s-v 間で漏えいが発生する場合とそうでない場合に分けて考える．s-v 間で漏えいが発生するならば，s と t は必ず分離される．漏えいが発生しないならば，s と v の縮約を行う．すなわち，グラフ $G' = (V', E')$　$(V' = V - \{s\}, E' = E - \{e_{sv}\})$ を考え，G' における点 v を新たな端子 s とみなし，G' の漏えい分布を $(p'_1, ..., p'_{|E|})$ とすると，以下のように再帰的に書ける．

$$\mathcal{E}_{\text{st-flow}}(G, p_1, ..., p_{|E|}) = p_{sv} + (1 - p_{sv})\mathcal{E}_{\text{st-flow}}(G', p'_1, ..., p'_{|E|})$$

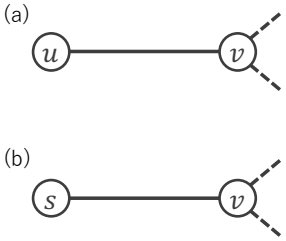

図 4　次数 1 の点のパターン

- 並列縮退
  図 5(a) のように，同じ端点を共有する 2 本の辺が存在する状況を考えよう．それぞれの漏えい確率を p'_{uv}，p''_{uv} とすると，この 2 辺で通信される情報が全て漏えいする確率は $p'_{uv}p''_{uv}$ である．この状況は図 5(b) のように 1 本の辺で表現できる．図 5(b) における p_{uv} の値として，図 5(a) における 2 辺で通信される情報が全て漏えいする確率を設定すれば良い．よって p_{uv} の値は，

$$p_{uv} = p'_{uv}p''_{uv}$$

とする．

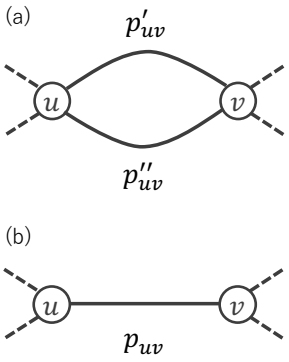

図 5　並列縮退の様子

- 直列縮退
  図 6(a) のように，$\deg(w) = 2$ の端子ではない点 w が存在する状況を考えよう．点 w と隣接する点 u，v 間の辺における漏えい確率を p_{uw}，p_{wv} とする．このとき，点 w を経由して u-v 間で通信される情報が漏えいする確率は，u-w 間と w-v 間の少なくとも片方で漏えいが発生する確率に等しいので，$1-(1-p_{uw})(1-p_{wv})$ である．並列縮退の場合と同様に，この状況は図 6(b) のように 1 本の辺で表現できる．図 6(b) における p_{uv} の値として，図 6(a) で点 w を経由して u-v 間で通信される情報が漏えいする確率を設定すれば良い．よって p_{uv} の値は，

$$p_{uv} = 1 - (1 - p_{uw})(1 - p_{wv})$$

とする．

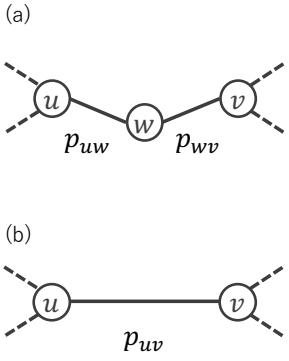

図 6　直列縮退の様子

ここまで説明した 3 つの縮退手法（次数 1 の点の除去，直列縮退，並列縮退）のみで漏えい確率が求められる場合もある．具体的には，図 3 の漏えい鍵共有直並列グラフ $(G^{\mathrm{ex2}}, p_{ab}, p_{ac}, ...)$ において，点 b，d をそれぞれ端子 s，t としたとき，上記の操作を実行不能になるまで行うことで，最終的に G^{ex2} を s，t の 2 点とそれを結ぶ 1 辺のみに縮退できる．そのとき，s-t 間の辺の漏えい確率 p_{st} が求める漏えい確率と一致している．

しかし，これらの 3 つの縮退手法のみでは漏えい確率が求められない場合もある．具体的には，G^{ex2} において，点 a，d をそれぞれ端子 s，t としたとき，グラフ内の点が s，t のみになるまで縮退を行うことができない．そのような場合は，Polygon-to-Chain 縮退 [9] を用いて，新たなグラフに縮退する必要がある．次の 4.3 節では，端子数が 2 の場合に特化した Polygon-to-Chain 縮退アルゴリズムを記述する．

4.3 $k = 2$ の場合に特化した Polygon-to-Chain 縮退

3 つの縮退手法（次数 1 縮退，直列縮退，並列縮退）を限界まで行ったとき，端子 s，t 以外にも点が存在している状況を考える．直列縮退により，s と t 以外の次数 2 の点は全て除去されるため，グラフ内の s と t 以外の点は全て次数が 3 以上であることに注意されたい．このとき，k 端子直並列グラフの形は 7 種類に限られることが示されている [9]．その内，明らかに 2 端子の場合に出現しない形を除くと図 7 のいずれかに限られる．さらに本研究では，2 端子に限定したとき，図 7(b)，(c)，(d) の形も出現しないことを証明する（黒点は端子を表す）．

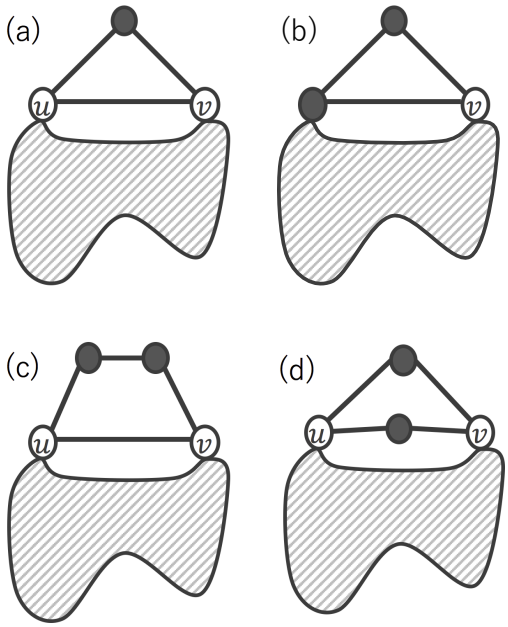

図 7 限界まで縮退した後のグラフのパターン

補題 1. 直並列グラフ G に対して *4.2* 節の縮退アルゴリズムを適用して最終的に得られるグラフ G' が，*2* 点 s，t とそれらを結ぶ辺からなるグラフではないとき，G' は図 *7(a)* の形状である．

証明． 直並列グラフの性質として，K_4-minor-free であることが知られている [4]．次数 1 縮退，直列縮退，並列縮退を限界まで行ったとき，図 7(b) の形の 2 端子直並列グラフが得られたと仮定する．このとき，s と t 以外の点は全て次数が 3 以上であるため，図 8 のようなマイナーが必ず存在する．しかし，図 8 のグラフはマイナーとして次数 4 の完全グラフを持つため，K_4-minor-free であることに反する．よって，図 7(b) の形は現れない．図 7(c) や (d) の場合についても同様に示せる． □

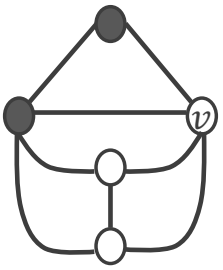

図 8 図 7(b) におけるマイナーの例

以上により，2 端子直並列グラフについては，図 7(a) の場合のみを考えれば良い．加えて，s と t 以外の端子は存在しないため，s，t はどちらも図 7(a) の形になっている．

- Polygon-to-Chain 縮退 [9]
  図 9(a) のように端子 s と s に隣接する 2 点が，3 点クリークになっている状況を考えよう．Polygon-to-Chain 縮退とは，図 9(a) のグラフ G から図 9(b) のグラフ G' に縮退する手法である．このとき，元のグラフの信頼度 $R(G, p_1, p_2, ..., p_{|E|})$ と得れれたグラフの信頼度 $R(G', p'_1, p'_2, ..., p'_{|E|})$ について，

$$R(G, p_1, p_2, ..., p_{|E|}) = \Omega R(G', p'_1, p'_2, ..., p'_{|E|}) \quad (2)$$

  を満たすような係数 Ω が存在する．Ω と各辺の漏えい確率 p'_{us}，p'_{sv} の値は，以下のようになる．

$$p'_{us} = \frac{\alpha}{\alpha + \delta}$$
$$p'_{sv} = \frac{\beta}{\beta + \delta}$$
$$\Omega = \frac{(\alpha + \delta)(\beta + \delta)}{\delta}$$

$$\alpha = p_{us}(1 - p_{sv})p_{uv}$$
$$\beta = (1 - p_{us})p_{sv}p_{uv}$$
$$\delta = (1 - p_{us})(1 - p_{sv})(1 - p_{uv})$$
$$\left(1 + \frac{p_{us}}{1 - p_{us}} + \frac{p_{sv}}{1 - p_{sv}} + \frac{p_{uv}}{1 - p_{uv}}\right)$$

よって，生成鍵 u の Eve への漏えい確率は，式 (1) 及び (2) により

$$\mathcal{E}_{\text{st-flow}}(G, p_1, ..., p_{|E|}) = 1 - \Omega + \Omega\mathcal{E}_{\text{st-flow}}(G', p'_1, ..., p'_{|E|})$$

と書ける．すなわち，$\mathcal{E}_{\text{st-flow}}(G', p'_1, ..., p'_{|E|})$ を再帰的に計算することで，u の漏えい確率が求められる．

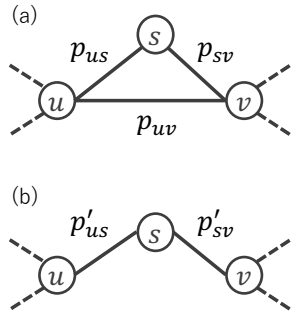

図 9 Polygon-to-Chain 縮退の様子

5. 漏えい確率の具体例

本節では，いくつかの漏えい鍵共有直並列グラフに対して 4 節で示したアルゴリズムにより，生成鍵 u の Eve への漏えい確率を求める．例として図 10，11，12，13 のような漏えい鍵共有直並列グラフ G^{ex3}，G^{ex4}，G^{ex5}，G^{ex6} を考え，各事前鍵は確率 p で独立に漏えいすると仮定すると，漏えい確率を与える多項式は以下のようになる．

$$\mathcal{E}_{\text{st-flow}}(G^{\text{ex3}}, p, ...) = p^2 + 4p^3 - 8p^4 + 5p^5 - p^6$$

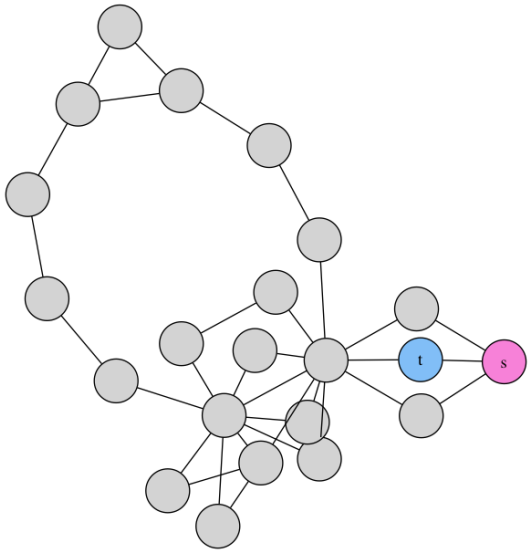

図 10 2 端子直並列グラフの例 G^{ex3}

$$\begin{aligned}\mathcal{E}_{\text{st-flow}}(G^{\text{ex4}}, p, ...) = {} & 8p^2 - 6p^3 - 23p^4 + 84p^5 - 162p^6 \\ & - 93p^7 + 1068p^8 - 1339p^9 \\ & - 1938p^{10} + 7368p^{11} - 6712p^{12} \\ & - 6173p^{13} + 25249p^{14} - 37009p^{15} \\ & + 34628p^{16} - 23049p^{17} + 11271p^{18} \\ & - 4053p^{19} + 1048p^{20} - 185p^{21} \\ & + 20p^{22} - p^{23}\end{aligned}$$

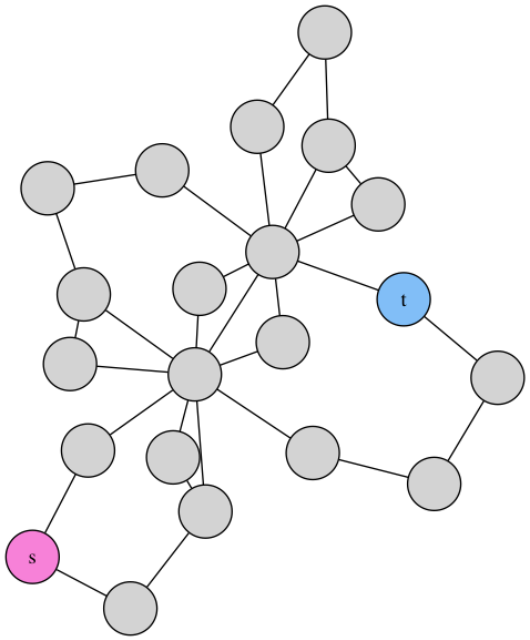

図 11 2 端子直並列グラフの例 G^{ex4}

$$\mathcal{E}_{\text{st-flow}}(G^{\text{ex5}}, p, ...) = \frac{f_5}{g_5}$$

$$\begin{aligned}f_5 = {} & 4p^2 + 14p^3 - 124p^4 - 196p^5 + 2265p^6 1953p^7 \\ & - 18112p^8 + 58371p^9 - 20701p^{10} - 280443p^{11} \\ & + 812634p^{12} - 909175p^{13} - 587646p^{14} + 4306461p^{15} \\ & - 9311388p^{16} + 13358335p^{17} - 14488358p^{18} \\ & + 12455864p^{19} - 8666063p^{20} + 4919046p^{21} \\ & - 2278126p^{22} + 855157p^{23} - 256669p^{24} + 60219p^{25} \\ & - 10650p^{26} + 1336p^{27} - 106p^{28} + 4p^{29}\end{aligned}$$

$$\begin{aligned}g_5 = {} & 1 - 18p^2 + 29p^3 + 86p^4 - 328p^5 + 278p^6 + 539p^7 \\ & - 1834p^8 + 2631p^9 - 2365p^{10} + 1442p^{11} - 601p^{12} \\ & + 165p^{13} - 27p^{14} + 2p^{15}\end{aligned}$$

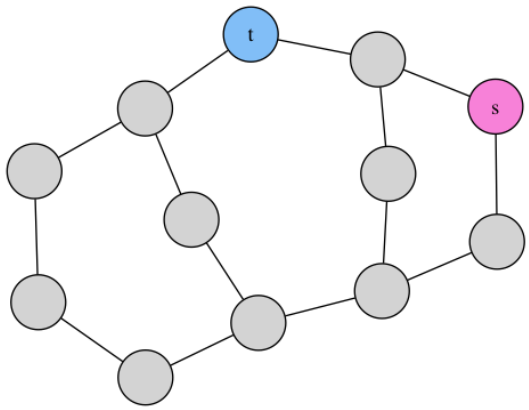

図 12　2 端子直並列グラフの例 G^{ex5}

$$\mathcal{E}_{\text{st-flow}}(G^{\mathrm{ex6}}, p, ...) = \frac{f_6}{g_6}$$

$$\begin{aligned}
f_6 = {} & 3p^2 + 3p^3 - 31p^4 - 21p^5 + 153p^6 - 17p^7 \\
& - 392p^8 + 702p^9 + 238p^{10} - 4356p^{11} + 3200p^{12} \\
& + 14432p^{13} - 20384p^{14} - 41280p^{15} + 84560p^{16} \\
& + 273040p^{17} + 53248p^{18} + 2009856p^{19} - 1005568p^{20} \\
& + 9638912p^{21} + 1961984p^{22} - 23961600p^{23} \\
& - 20840448p^{24} - 128548864p^{25} + 43581440p^{26} \\
& - 125304832p^{27} - 120848384p^{28} - 741605376p^{29} \\
& + 1007157248p^{30} - 1518338048p^{31} - 191889408p^{32} \\
& + 2385510400p^{33} + 794820608p^{34} + 517996544p^{35} \\
& + 5303697408p^{36} - 784334848p^{37} - 10120855552p^{38} \\
& + 12329156608p^{39} + 10001317888p^{40} \\
& - 59598962688p^{41} + 90006618112p^{42} \\
& - 53545533440p^{43} - 35676749824p^{44} \\
& + 139007098880p^{45} - 342948052992p^{46} \\
& + 909013942272p^{47} - 1969580670976p^{48} \\
& + 3021029179392p^{49} - 2802108235776p^{50} \\
& - 17783586816p^{51} + 5712191594496p^{52} \\
& - 12904424435712p^{53} + 19141582949376p^{54} \\
& - 22265603999744p^{55} + 21548523661568p^{56} \\
& - 17841930612224p^{57} + 12834507864944p^{58} \\
& - 8093770776720p^{59} + 4498243225360p^{60} \\
& - 2209192905276p^{61} + 959530958140p^{62} \\
& - 368249124520p^{63} + 124576730448p^{64} \\
& - 37001143276p^{65} + 9593706148p^{66} \\
& - 2154542192p^{67} + 414741640p^{68} - 67479256p^{69} \\
& + 9104440p^{70} - 991744p^{71} + 83836p^{72} - 5162p^{73} \\
& + 206p^{74} - 4p^{75}
\end{aligned}$$

$$\begin{aligned}
g_6 = {} & 1 - 7p^2 - 1p^3 + 25p^4 - 14p^5 - 21p^6 + 84p^7 - 67p^8 \\
& - 455p^9 + 620p^{10} + 1415p^{11} - 2983p^{12} - 1730p^{13} \\
& + 10280p^{14} - 8175p^{15} - 20200p^{16} + 56320p^{17} \\
& - 23598p^{18} - 86252p^{19} + 45972p^{20} + 319958p^{21} \\
& - 525198p^{22} - 296288p^{23} + 1789472p^{24} - 1481432p^{25} \\
& - 2653624p^{26} + 7251120p^{27} - 3818640p^{28} \\
& - 10410504p^{29} + 24416312p^{30} - 30829408p^{31} \\
& + 54813088p^{32} - 135317964p^{33} + 241894178p^{34} \\
& - 223606286p^{35} - 98621072p^{36} + 745372221p^{37} \\
& - 1494062025p^{38} + 2010990350p^{39} - 2085641546p^{40} \\
& + 1753365208p^{41} - 1224017096p^{42} + 718331926p^{43} \\
& - 356374292p^{44} + 149599036p^{45} - 52979384p^{46} \\
& + 15721630p^{47} - 3866034p^{48} + 774692p^{49} \\
& - 123388p^{50} + 15038p^{51} - 1318p^{52} \\
& + 74p^{53} - 2p^{54}
\end{aligned}$$

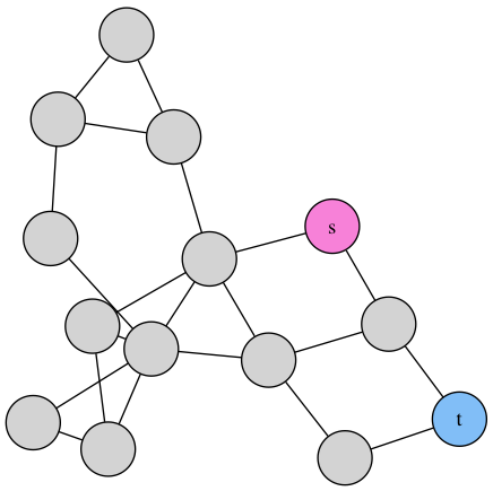

図 13　2 端子直並列グラフの例 G^{ex6}

6. おわりに

入力として漏えい鍵共有グラフ $(G, p_1, p_2, ..., p_{|E|})$ と二人のプレイヤー s，t が与えられたとき，st-フロープロトコルを実行した場合における生成鍵 u の Eve への漏えい確率 $\mathcal{E}_{\text{st-flow}}(G, p_1, p_2, ..., p_{|E|})$ を求める問題は，2 端子間ネットワーク信頼度の問題と同値であり，#p 完全であることを述べた．また，k 端子直並列グラフにおける信頼度問題を解決する既存のアルゴリズムをカスタマイズし，漏えい鍵共有直並列グラフにおける漏えい確率 $\mathcal{E}_{\text{st-flow}}(G, p_1, p_2, ..., p_{|E|})$ を求める線形時間アルゴリズムを記述した．更に，いくつかの漏えい鍵共有直並列グラフに対して具体的に漏えい確率を与える多項式を示した．

参考文献

[1] Bennett, C.H., Brassard, G., Crepeau, C., Maurer, U.M.: Generalized privacy amplification. IEEE Transactions on Information Theory 41(6), 1915–1923 (Nov 1995)

[2] Colbourn, C.J., Colbourn, C.: The combinatorics of network reliability, vol. 200. Oxford University Press New York (1987)

[3] Csiszar, I., Narayan, P.: Secrecy capacities for multiple terminals. IEEE Transactions on Information Theory 50(12), 3047–3061 (Dec 2004)

[4] Duffin, R.: Topology of series-parallel networks. Journal of Mathematical Analysis and Applications 10(2), 303 – 318 (1965)

[5] Mizuki, T., Nakayama, S., Sone, H.: An application of st-numbering to secret key agreement. International Journal of Foundations of Computer Science 22(05), 1211–1227 (2011)

[6] Nagaraja, S.: Privacy Amplification with Social Networks, pp. 58–73. Springer Berlin Heidelberg, Berlin, Heidelberg (2010)

[7] Ostádal, R., Svenda, P., Matyás, V.: A new approach to secrecy amplification in partially compromised networks (invited paper). In: Security, Privacy, and Applied Cryptography Engineering - 4th International Conference, SPACE 2014, Pune, India, October 18-22, 2014. Proceedings. pp. 92–109

[8] Provan, J.S., Ball, M.O.: The complexity of counting cuts and of computing the probability that a graph is connected. SIAM J. Comput. 12(4), 777–788 (1983)

[9] Satyanarayana, A., Wood, R.K.: A liner-time algorithm for computing k-terminal reliability in series-parallel networks. SIAM J. Comput. 14(4), 818–832 (1985)

[10] Shannon, C.: Communication theory of secrecy systems. Bell System Technical Journal, Vol 28, pp. 656715 (Oktober 1949)

[11] Watanabe, S., Matsumoto, R., Uyematsu, T.: Strongly secure privacy amplification cannot be obtained by encoder of slepian-wolf code. IEICE Transactions on Fundamentals of Electronics, Communications and Computer Sciences E93.A(9), 1650–1659 (2010)

[12] 佐々木達也, 林優一, 水木敬明, 曽根秀昭: 一様漏えい鍵共有完全二部グラフに関する一考察. 2017 年電子情報通信学会総合大会，A-7-3 (2017)

[13] 増田真吾, 林優一, 水木敬明, 曽根秀昭: 漏えい鍵共有グラフにおける効果的な鍵選択に関する考察. コンピュータセキュリティシンポジウム 2016， pp. 1276–1283 (2016)