グラフレイアウト問題に対する固定パラメータと厳密指数時間アルゴリズム

Fixed Parameter and Exact Exponential Time Algorithms for Graph Layout Problems

小林靖明

指導教員：玉木久夫 教授

2013 年 12 月

概要

グラフレイアウト問題は，n 頂点グラフの頂点集合の分割 $(V_1, V_2, \ldots, V_t)$ において，各 V_i 上の順列 π_i によって定まる目的関数 $f(\pi_1, \pi_1, \ldots, \pi_t)$ を最適化する問題である．本学位請求論文では，２部グラフの階層描画における辺交差数最小化問題とグラフのパス幅を計算するアルゴリズムに関する研究結果をまとめる．

２部グラフの交差数最小化問題のひとつとして知られている，One-Sided Crossing Minimization(OSCM) について，$O(k2^{\sqrt{2k}} + n)$ 時間で動作する準指数時間固定パラメータアルゴリズムを与えた．ここでパラメータ k は辺交差数とする．この結果は Fernau らの結果の実行時間 $2^{O(\sqrt{k}\log k)} + n^{O(1)}$ を改善している．さらに，このアルゴリズムの実行時間の指数部が指数時間仮説のもとで最適であることを示した．

もうひとつの交差数最小化問題として，Two-Layer Crossing Miniza-tion(TLCM) を考える．TLCM においては，多項式サイズのカーネル化を与えた．カーネル化とはインスタンスとパラメータが与えられたときに判定問題の答えを変えることなく，インスタンスの大きさをパラメータにのみ依存する大きさへ圧縮する多項式時間の前処理のことである．この結果の系として，TLCM の $2^{O(k\log k)} + O(n)$ 時間で動作する固定パラメータアルゴリズムを与えた．この結果は既存のアルゴリズムの実行時間 $2^{O(k^3)}n$ を大幅に改善した．

上記の 2 つの問題とは異なる有向グラフのパス幅を求める問題を考える．有向グラフのパス幅は頂点の順列によって特徴付けられるので，グラフレイアウト問題として定式化できる．本論文では，この問題に対して $O(1.89^n)$ 時間アルゴリズムを与えた．この結果の特別な場合として，Suchan と Villanger の無向グラフのパス幅を求める $O(1.9657^n)$ 時間アルゴリズムの実行時間を改善した．

目 次

第 1 章 はじめに 5

1.1 諸定義 5

1.2 グラフレイアウト問題 7

1.2.1 ２層描画 7

1.2.2 パス幅 9

1.3 NP 困難問題に対するアプローチ 13

1.3.1 パラメータ化計算量理論 13

1.3.2 厳密指数時間アルゴリズム 15

1.4 本論文の構成 16

第 2 章 One-sided crossing minimization 17

2.1 研究背景 17

2.2 準指数時間固定パラメータアルゴリズム 20

2.2.1 諸定義 20

2.2.2 区間表現 22

2.2.3 交差数を計算する線形時間アルゴリズム 25

2.2.4 区間表現上の動的計画法 26

2.2.5 アルゴリズムの全体像 29

2.3 計算量の下界 31

2.4 まとめ 33

第 3 章 Two-layer crossing minimization 35

3.1 関連研究 36

3.2 諸定義 37

3.3 カーネル化 38

3.3.1 TLCM に対するカーネル化 38

3.3.2 LEW-TLCM に対するカーネル化 48

3.3.3 アルゴリズムの全体像 49

3.4 まとめ 52

第 4 章　有向グラフのパス幅を計算するアルゴリズム　54
4.1　研究背景 54
4.1.1　パス幅を用いた最大独立集合問題の解法 54
4.1.2　パス幅とグラフ描画の関係 56
4.1.3　先行研究 57
4.2　準備 58
4.3　$2^n n^{O(1)}$ 時間アルゴリズム 63
4.4　$O(1.89^n)$ 時間アルゴリズム 64
4.4.1　アルゴリズム Large-width 64
4.4.2　XP アルゴリズム 66
4.4.3　アルゴリズム Small-width 68
4.4.4　アルゴリズムの全体像 72
4.5　まとめ 72

参考文献　74

図 目 次

1.1 ２層描画の例 8
1.2 無向グラフのパス分解と木分解の例 10
1.3 有向グラフのパス分解と順列例 12

2.1 多層描画と OSCM 19
2.2 素朴な区間表現の例 22
2.3 摂動された区間表現の例 24
2.4 $c(u,v)$ を計算する式 25

3.1 補題 3.1 の例 39
3.2 P_i と P_i' の例 40
3.3 補題 3.2 のタイトな例 42
3.4 補題 3.4 における $\mathcal{D}$ と $\mathcal{D}'$ の例 44
3.5 補題 3.4 の例 44
3.6 $\mathcal{D}$ と $\mathcal{D}'$ の例 45
3.7 LEW-TLCM のインスタンス (G,w) と $\mathrm{unfold}(G,w)$ 48
3.8 $T_{\mathcal{B}}$ の例 51

4.1 キャタピラーとその２層描画の例 57
4.2 $\mathrm{sat}(U)$ の例 60
4.3 $\mathrm{push}_4(U)$ の例 61

第1章　はじめに

グラフの概念は，いくつかのオブジェクトを頂点集合として表し，オブジェクト間の関係を頂点同士結ぶ辺として表現する数学的概念である．現実世界の多くの対象をグラフとしてモデル化することができ，その上で最適化問題を解くことは現実世界へ多くの応用を持つ．このためにグラフ上の最適化問題に対する研究は広く行われている．しかしながら，そのような問題の中には，その問題を解く多項式時間アルゴリズムが知られていないものが存在し，さらには，ある計算量理論の仮定の下で多項式時間アルゴリズムが存在しないような問題がいくつも知られている．現実世界への応用という観点から，そのような問題に対するできるだけ高速なアルゴリズムの設計をする研究が盛んに行われている．

本論文では，２つのグラフ描画問題と多くのグラフ上の最適化問題を解くために重要とされている，グラフの “幅” を求める問題について考える．これらの問題は**グラフレイアウト問題**として定式化することができるが，いずれも NP 困難である．

P = NP でない限り，NP 困難な問題に対する多項式時間アルゴリズムは存在しないが，その一方で，そのような “難しい” 問題に対するアプローチは盛んに行われている．本論文では指数時間で動作し，厳密解を求めるアルゴリズムを提案する．これらのアルゴリズムは，ある程度の大きさのインスタンスに対して許容できる時間で動作したり，特殊なケースにおいては高速に動作したりする．グラフのレイアウト問題に対して，これらのアプローチを適用し，本結果のアルゴリズムの優位性を理論的と実用的な観点から議論する．

1.1　諸定義

本節では，本論文で用いられる用語や記法について説明するが，いくつかのグラフ理論や計算量理論，アルゴリズム論などの概念については読者

が習得しているものと仮定している．これらの詳細については [34, 56, 31] などを参照して頂きたい．

無向グラフ : 無向グラフ G において，その頂点集合を $V(G)$ と表し，辺集合を $E(G)$ で表す．本論文では，グラフは常に単純（多重辺および自己ループを持たない）であるとする．頂点 v の**次数** $d(v)$ とは，v に接続される辺の数のことである．頂点 $v \in V(G)$ の**隣接点集合** $\{w \in V(G) : \{v, w\} \in E(G)\}$ を $N(u)$ で表す．単純グラフの場合は，頂点の隣接点集合の個数とその次数が一致する．隣接点集合の定義は以下のように頂点集合上へ拡張できる．$U \subseteq V(G)$ において，$N(U)$ を $\bigcup_{v \in U} N(v) \setminus U$ と表す．また，$G[U]$ を U によって**誘導される部分グラフ**，つまり $G[U] = (U, \{\{u, v\} \in E(G) : u, v \in U\})$，と記述する．

有向グラフ : 小節 1.2.2，4 章においては有向グラフについて扱う．用語や記法については，無向グラフと異なる部分のみ説明する．

有向グラフを G とする．有向グラフが**単純**であるとは，自己ループおよび，ある 2 点 $u, v \in V(G)$ において 2 つの同一方向の有向辺 (u, v) を持たないことである．単純有向グラフは辺 (u, v) と (v, u) を同時に持つことはあり得ることに注意する．無向グラフと同様に本論文では有向グラフは常に単純であるとする．本論文では，各頂点 $v \in V(G)$ において，**入隣接点集合** $N^{-}(v) = \{w \in V(G) : (v, w) \in E(G)\}$ のみを考える．(出隣接点集合も同様に定義できるが，本論文では用いない．) また，無向グラフと同様に $U \subseteq V(G)$ において $N^{-}(U) = \bigcup_{v \in U} N^{-}(v) \setminus U$ と定義する．

２部グラフ:無向グラフ G が**２部グラフ**であるとは，$V(G)$ の 2 分割 $X, Y \subseteq V(G), X \cap Y = \emptyset$ が存在し，$G[X]$ および $G[Y]$ がそれぞれ辺を持たないときである．グラフ G が 2 部グラフであるとき，$V(G)$ のある 2 分割を $(X(G), Y(G))$ で表し，G を $(X(G), Y(G), E(G))$ と表す．また，本論文において 2 部グラフを扱うときには，ある 2 分割がすでに与えられているものと仮定する．

頂点順列と順序 : $X \subseteq V(G)$ とする．X 上の**順列**とは，長さ $|X|$ の列で，X の各要素がその列のなかでちょうど 1 回現れるもののことである．また，X が特に指定されていないときの順列を単に**列**と呼ぶ．X 上の順列全体を $\Pi(X)$ で表す．$|X| = \{v\}$ のとき，X の唯一の順列を v と書くことがある．$\pi \in \Pi(X)$ において，$|\pi|$ は π の長さ，つまり $|\pi| = |X|$ とし，$V(\pi) = X$ と定義する．G の順列および G が持つ順列というときは，暗

黙のうちに $V(G)$ 上の順列を指している．X の順序関係 $<$ と $u, v \in X$ において，$u < v$ または $u = v$ を満たすとき，$u \leq v$ と記述する．$\pi \in \Pi(X)$ としたとき，$<_\pi$ を π によって得られる全順序と定義する．つまり，任意の異なる $u, v \in X$ において $u <_\pi v$ であることと，u が π の中で v より前に現れることが一致する．

1.2　グラフレイアウト問題

グラフレイアウト問題は，以下のように定義できる．無向または有向グラフ $G = (V(G), E(G))$ において，$V(G)$ の分割を $V_1, V_2, \ldots, V_t$ とし，各 $1 \leq i \leq t$ において，$\pi_i \in P(V_i)$ とする．このとき，ある関数値 $f(\pi_1, \pi_2, \ldots, \pi_t)$ を最小化，または最大化する問題を t **グラフレイアウト問題**と呼ぶ．一般的に，$t = 1$ の場合がグラフレイアウト問題 [33] としてよく研究されているが，本論文では $t = 1, 2$ の場合の具体的な問題，グラフの２層描画交差数最小化問題と有向グラフのパス幅問題についての結果を述べる．それぞれの正確な定義は次の２つの小節に記す．

1.2.1　２層描画

Purchase[96] によると，グラフ描画の可視性において辺の交差は最も重要な役割を果たしていると言われている．そのため，辺交差数の最小化問題はグラフ描画の分野で最もよく研究されているもののひとつである [99]．グラフの描画方法が異なれば辺交差数最小化問題は別の最適化問題として定式化される．本論文ではグラフを階層的に描画する方法，特に２部グラフを２つの層に描画する問題に関して取り組む．

２部グラフ G の**２層描画**とは $X(G)$ に含まれる頂点を直線 (X **層**) 上に配置，$Y(G)$ に含まれる頂点を X 層と平行な直線 (Y **層**) 上に配置して，各辺を直線分で描画する方法である．ただし頂点同士が重なってはいけない (図 1.1)．２層描画はデータの可視化 [6, 71]，遺伝子マッピング [111]，VLSI レイアウト [83] などに応用を持ち，グラフ描画の世界で盛んに研究されている問題のひとつである．２層描画における**交差**とは，２つの辺が端点以外で互いに交わるとき，その交点のことである．辺交差数を最小化する問題を考えるために，以下の概念を導入する．G の**組合せ的２層描画** $\mathcal{D}$ とは，G と $X(G)$ 上の全順序 $<_X$，$Y(G)$ 上の全順序 $<_Y$ の三つ組み $(G, <_X, <_Y)$ と定義する．G の組合せ的２層描画 $\mathcal{D}$ における**交差**と

は，G の辺対 $(\{u,v\},\{x,y\})$ で $x <_X u$ かつ $v <_Y y$ を満たすものである．また，$\mathcal{D}$ における交差数を $\mathrm{bcr}(\mathcal{D})$ で表す．つまり，

$$\mathrm{bcr}(\mathcal{D}) = \sum_{\{u,v\}\in E(G)} |\{\{x,y\} \in E(G) : x <_X u, v <_Y y\}|$$

である．G のある２層描画が与えられたときに，以下のように組合せ的２層描画を定義できる：各層を数直線とみなすと，描画された頂点の座標の大小は全順序関係を満たし，その全順序に従って $(G, <_X, <_Y)$ を構成する．

組合せ的２層描画の概念を用いると以下が成り立つ．

補題 1.1. 任意の２つの２層描画が組合せ的に同値，つまり，同じ組合せ的２層描画をもつとき，その辺交差数は一致する．

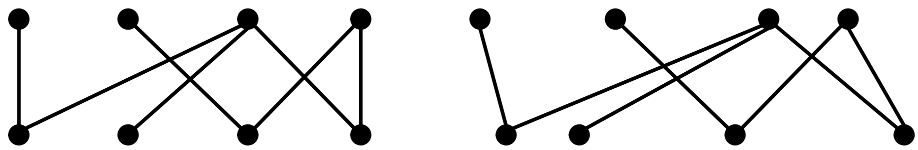

図 1.1: 同一の組合せ的２層描画をもつ２層描画の例．組合せ的２層描画が一致すれば交差数も一致する．

本論文では，交差数最小の組合せ的２層描画を求める問題に取り組む．以降で２層描画について言及する場合は，組合せ的２層描画を指す．本論文では以下の２つの交差数最小化問題に関して考える．

One-Sided Crossing Minimization (OSCM) は，２部グラフ G と $X(G)$ 上の全順序 $<_X$，正整数 k が与えられたとき，$\mathrm{bcr}(\mathcal{D} = (G, <_X, <_Y)) \leq k$ を満たすような $Y(G)$ 上の全順序 $<_Y$ が存在するかを判定する問題である．OSCM は２グラフレイアウト問題において，ある一つの順列が固定された問題として定式化できる．また，Sugiyama-Tagawa-Toda[108] は OSCM を等価なフィードバック辺集合問題として定式化した．この問題は１グラフレイアウト問題としても見ることができる．

OSCM における先行研究や応用については２章で述べる．本論文では，この問題に対する準指数時間固定パラメータアルゴリズムと指数時間仮説に基づく計算量の下界を示す．具体的には，OSCM を解く $O(k2^{\sqrt{2k}} + n)$ 時間アルゴリズムを与える．ここで，n は G の頂点数とする．さらに，指

数時間仮説が成り立つならばOSCMに対する$2^{o(\sqrt{k})}n^{O(1)}$時間アルゴリズムが存在しないことを示す．固定パラメータアルゴリズムの定義は1.3節で述べ，指数時間仮説については2章で述べる．

Two-Layer Crossing Minimization (TLCM) は，2部グラフGと正整数kが与えられたとき，$\mathrm{bcr}(\mathcal{D}=(G,<_X,<_Y))\le k$であるような$X(G),Y(G)$上の全順序$<_X,<_Y$が存在するかを判定する問題である (いくつかの論文ではTLCMの代わりに，Two-Sided Crossing Minimizationといわれることがある)．OSCMとの違いは$X(G)$上の順序が入力として与えられるかどうかであり，OSCMと同様にTLCMも2グラフレイアウト問題として定式化できる．この問題における関連研究については3章で述べる．この問題に対するカーネル化アルゴリズムとそれに基づいて得られる固定パラメータアルゴリズムを3章で示す．具体的には，TLCMのインスタンス(G,k)が与えられたとき，(G,k)と等価なインスタンス(H,k)を求める多項式時間アルゴリズムを与える．ここで，等価なインスタンスとは(G,k)がYESインスタンスであるとき，かつそのときのみ(H,k)がYESインスタンスになることである．またカーネル化の詳しい定義は1.3節で述べる．

k交差以下で描画できるグラフの疎性に関して以下の補題が成り立ち，2章と3章においてを利用する．

補題 1.2. *2部グラフGが辺交差数k以下の2層描画を持つならば，$|E(G)|\le|V(G)|+k-1$を満たす．*

証明. kの帰納法で証明する．$k=0$の場合，Gは森でなければならないので明らかに成り立つ（グラフに閉路が存在する場合明らかに辺交差が生じる.）．$k>0$と仮定し，交差数kの2層描画を考える．eをこの描画において少なくとも一つの交差を生じさせている辺とする．$G-e$をGから辺eを取り除いたグラフとすると，$G-e$は辺交差数$k-1$以下の2層描画を持つ．帰納法の仮定より，$|E(G-e)|\le|V(G-e)|+(k-1)-1$であり，$|E(G)|-1\le|V(G)|+(k-1)-1$なので，補題が成り立つ．□

1.2.2　パス幅

パス幅の概念はRobertson-Seymour[97]によって導入され，木幅の概念とともに，グラフマイナー理論[97]やグラフアルゴリズムのなかで重要な役割を果たしている．パス幅や木幅の応用についてはBodlaenderの論文[14]が詳しい．

無向グラフ G の**パス分解**とは，$V(G)$ の部分集合を頂点と見なしたパス $P = (X_1, X_2, \dots, X_t)$ で以下を満たす：

1 $\bigcup_{1 \le i \le t} X_i = V(G)$,

2 各 $\{u, v\} \in E(G)$ において，$\{u, v\} \subseteq X_i$ を満たす添字 i $(1 \le i \le t)$ が存在する，

3 各 $v \in V(G)$ において，v を含む X_i は P 上で連結になる．

パス分解 P の**幅**は $\max_{1 \le i \le t} |X_i| - 1$ と定義され，G の**パス幅** $\mathrm{pw}(G)$ とは，G が幅 k のパス分解を持つような最小の k として定義される．

G の木分解とは P をパスから木に緩和し，上記の１，２，３の条件を満たしたものである．また木分解の幅や木幅も，パス分解の幅やパス幅と同様に定義することができる．グラフ G のパス分解は G の木分解でもある．図 1.2 は，同一のグラフのパス分解と木分解の例である．

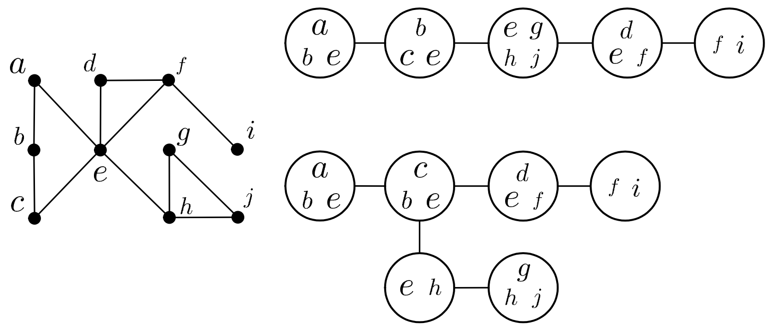

図 1.2: 無向グラフのパス分解と木分解の例

パス分解を用いた動的計画法の設計の例やグラフ描画との関連性を 4 章で述べる．

有向グラフのパス分解の定義は無向グラフと場合と類似している．有向グラフ $G = (V(G), E(G))$ のパス分解とは，パス $P = (X_1, X_2, \dots, X_t)$ で無向グラフのパス分解の条件１と３と以下を満たす：

2’ 各 $(u, v) \in E(G)$ において，$u \in X_i$ かつ $v \in X_j$ を満たす添字 i, j $(1 \le i \le j \le t)$ が存在する．

有向グラフのパス分解の幅およびパス幅は無向グラフと同様に定義できる．Barát[5] によると，有向グラフのパス幅の概念はReed-Seymour-Thomasによって１９９５年頃導入された．

有向グラフのパス幅を求める問題は，無向グラフのパス幅を求める問題の一般化と見なせる．この事実は以下の補題によって確認できる．

補題 1.3. 任意の無向グラフ $G = (V(G), E(G))$ について，G の各辺 $\{u, v\} \in E(G)$ を２つの有向辺 $(u, v), (v, u)$ で置き換えた有向グラフを $\hat{G} = (V(G), \{(u, v), (v, u) : \{u, v\} \in E(G)\})$ としたとき，$\mathrm{pw}(G) = \mathrm{pw}(\hat{G})$ である．

証明. 有向グラフ $\hat{G}$ のパス分解 $P = (X_1, X_2, \ldots, X_t)$ において，条件 2' を有向辺 (u, v) および (v, u) で満たそうとすると，ある $\{u, v\} \subseteq X_i$ を満たす添字 i が存在しなければならない．これは無向グラフのパス分解の条件 2 と一致する．また G のパス分解は $\hat{G}$ のパス分解でもあるので，補題が成り立つ． □

無向および有向グラフのパス幅を求める問題は，以下のように１グラフレイアウト問題として特徴付けることができる．

無向 (有向) グラフ G において，$V(G)$ 上の順列 $\pi \in \Pi(V(G))$ を考える．各 i $(1 \leq i \leq |V(G)|)$ において，π_i を π の先頭から i 要素で構成される部分列とする．k を整数とし，すべての i $(1 \leq i \leq |V(G)|)$ において，$|N(V(\pi_i))| \leq k$ $(|N^-(V(\pi_i))| \leq k)$ を満たすとき，π を G に関して k **分離可能**な順列であるといい，G が k 分離可能な順列を持つような最小の k を G の**頂点分離数** $\mathrm{vs}(G)$ と呼ぶ．

無向および有向グラフのパス幅と頂点分離数に関して以下の結果が知られている．自己完結性のために G が有向グラフの場合を示す．

定理 1.1 (無向グラフ [73]，有向グラフ [114])**.** 任意の無向および有向グラフ G において，$\mathrm{pw}(G) = \mathrm{vs}(G)$.

証明. $n = |V(G)|$ とする．まず $\mathrm{pw}(G) \leq vs(G)$ を示す．$\pi = (v_1, v_2, \ldots, v_n)$ を $V(G)$ 上の $\mathrm{vs}(G)$ 分離可能な順列であるとする．$P = (X_1, X_2, \ldots, X_n)$ を以下のように構成する：各 i $(1 \leq i \leq n)$ において $X_i = \{v_i\} \cup N^-(V(\pi_i))$ とする．このとき，$\max_{1 \leq i \leq n} |X_i| - 1 \leq \mathrm{vs}(G)$ であるので，P がパス分解の条件を満たすことを示せば，$\mathrm{pw}(G) \leq \mathrm{vs}(G)$ を示せる．条件１については明らかである．

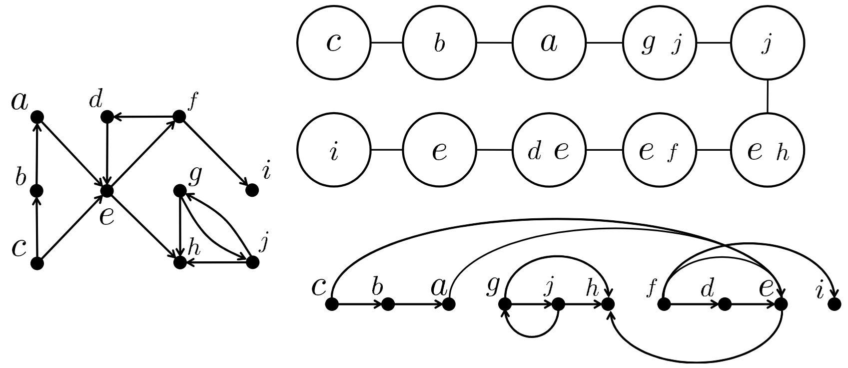

図 1.3: 有向グラフのパス分解と順列の例：幅 1 のパス分解と 1 分離可能な順列

条件 2'：$(v_i, v_j) \in E(G)$ とする．$v_i \in X_k$ を満たす最小の k を l_i とし，$v_j \in X_k$ を満たす最大の k を r_j とする．$l_i \le r_j$ とすると，条件 2' を満たすので，$l_i > r_j$ と仮定する．P の構成方法より，$v_j <_\pi v_i$ である。よって $v_i \in N^-(V(\pi_j))$ であり，$\{v_i, v_j\} \subseteq X_j$ なので矛盾する．
条件 3 : $v_i \in V(G)$ とする．$v_i \in N^-(V(\pi_j))$ を満たす最小添字 j を l_i とすると，明らかに $l_i \le k < i$ なる k において $v_i \in N^-(V(\pi_k))$ であり，$i < k$ なる k においては $v_i \notin N^-(V(\pi_k))$ なので，v_i を含む X_k は P 上で連結である．

よって $\mathrm{pw}(G) \le \mathrm{vs}(G)$ である．

次に $\mathrm{pw}(G) \ge \mathrm{vs}(G)$ を示す．$P = (X_1, X_2, \ldots, X_t)$ を G のパス分解で $\max_{1 \le i \le t} |X_i| - 1 = \mathrm{pw}(G)$ となるものとする．$V(G)$ 上の順列を以下のように構成する．各 $v \in V(G)$ において，$v \in X_i$ を満たす最大の i を r_v とする．順列 $\pi = (v_1, v_2, \ldots v_n)$ を $r_u < r_v$ ならば，$u <_\pi v$ を満たし，$r_u = r_v$ ならば，任意に順序付けて得られる順列とする．このとき任意の i $(1 \le i \le n)$ において，$N^-(V(\pi_i))$ に含まれる任意の頂点 v は $r_{v_i} \le r_v$ を満たし，$(v, v_i) \in E(G)$ より，ある k が存在して，$\{v_i, v\} \subseteq X_k$ を満たす．パス分解の条件 3 と区間集合のヘリーの定理より，$(\{v_i\} \cup N^-(V(\pi_i))) \subseteq X_j$ を満たす j が存在するので，$\mathrm{pw}(G) \ge \mathrm{vs}(G)$ である．

よって，$\mathrm{pw}(G) = \mathrm{vs}(G)$ である． □

この補題により，パス幅を求める問題は 1 グラフレイアウト問題として定式化できる (他にも無向グラフのパス幅は区間グラフを用いた特徴付け [16] やゲームを用いた特徴付け [74] 知られており，有向グラフのパス

幅もゲームを用いた特徴付けの研究が行われている [5, 114])．本論文では，有向グラフの頂点分離数を求める厳密指数時間アルゴリズムを与える．具体的には，n 頂点有向グラフが与えられたとき，頂点分離数を求める $O(1.89^n)$ 時間アルゴリズムを与える．無向および有向グラフのパス幅，頂点分離数の関連研究と応用に関しては 4 章で述べる．

1.3　NP 困難問題に対するアプローチ

P = NP でない限り NP 困難問題に対する多項式時間厳密アルゴリズムは存在しない．しかしながら，現実の問題をグラフ上の最適化問題としてモデル化したとき，その問題が NP 困難であることは珍しくない．この事実により多くの NP 困難問題が様々なアプローチを用いて研究されている：例えば，近似アルゴリズム [112] や乱択アルゴリズム [88]，ヒューリスティックス [85] などがある．本研究ではパラメータ化計算量理論と厳密指数時間アルゴリズムのアプローチによって前節で述べた問題に取り組む.

1.3.1　パラメータ化計算量理論

近年，**パラメータ化計算量理論** [35, 48, 94] は NP 困難な問題へのアプローチとして，最も盛んな研究分野のひとつとして認識されている．以下ではこの理論へのごく簡単な導入を与える．

Σ を有限のアルファベット集合とする．このとき，$L \subseteq \Sigma^* \times \mathbb{N}$ を**パラメータ化問題**と呼ぶ．パラメータ化問題 L が**固定パラメータ容易** (*Fixed Parameter Tractable, FPT*) であるとは，L のインスタンス $(I, k) \in \Sigma^* \times \mathbb{N}$ において $(I, k) \in L$ であるかを判定するアルゴリズムで，その実行時間が $f(k)|I|^{O(1)}$ であるものが存在することである．ここで，f は k に依存する任意の関数とし，$|I|$ はインスタンス I の大きさとする．さらに，そのアルゴリズムを**固定パラメータアルゴリズム**と呼ぶ.

固定パラメータアルゴリズムを考える最も初歩的な動機は，“パラメータ k の値が十分に小さいとき，固定パラメータアルゴリズムは高速に動作する” ということである．特にパラメータが定数のとき，そのアルゴリズムは多項式時間で動作する．さらに，考えている問題が現実的制約などからパラメータが小さいと仮定できることは少なくない．例えば，本論文のグラフ描画問題は視認性の限界という観点から，最適化対象のイ

ンスタンスは十分に疎で，辺交差数はあまり大きくならないと仮定できる [41]．よって，辺交差数をパラメータとして固定パラメータアルゴリズムを設計することは有意義である．他の例では，道路ネットワークをグラフとしてモデル化すると，グラフの次数は小さなパラメータとみなせたり，施設配置問題では，設立できる施設数は予算などの観点から余り多くの施設を開設することはできないと仮定できる．このように，パラメータが小さい場合に高速に動作するアルゴリズムが有用な場面は多く存在する．

また，パラメータ化計算量理論で問題を考えるもう一つの動機は問題の "困難性" である．NP 困難性の議論で得られるひとつの結論は "P = NP でない限り NP 困難問題に対する多項式時間アルゴリズムが存在しない" ということであった．その一方で，いくつかのパラメータ化問題はパラメータが定数であるときには多項式時間アルゴリズムを持つことがある．例えば，n 頂点グラフと整数 k が与えられたとき，大きさ k 以下の点カバーを持つか判定する問題や大きさ k 以上の独立集合を持つかを判定する問題は，k が定数のとき自明な多項式時間アルゴリズムが存在する．しかしながら，前者は FPT であることが知られており，後者に対する固定パラメータアルゴリズムは見つかってはいない．よって問題が FPT であるかどうかを考えるためには新しい計算量理論が必要である．

パラメータ化問題に対して固定パラメータアルゴリズム設計するテクニックはいくつも知られているが，その中でも最も強力なもののひとつにカーネル化がある．$L \subseteq \Sigma^* \times \mathbb{N}$ をパラメータ化問題とし，(I, k) を L のインスタンスとする．L に対する**カーネル化**とは，

- $(I, k) \in L \Leftrightarrow (I', k') \in L$,
- $|I'| \leq g(k)$，g はある計算可能関数,
- $k' \leq h(k)$，h は任意の関数,

を満たす L のインスタンス (I', k') を出力するアルゴリズムで，$|I| + k$ の多項式時間で動作するもののことである．ここで，(I', k') を L の**カーネル**と呼び，特に，g と h が多項式関数であるとき，それぞれを**多項式カーネル化**と**多項式カーネル**と呼ぶ．カーネル化のテクニックに関しては [59] が詳しい．

FPT とカーネル化に関して以下の定理が成り立つ．

定理 1.2 (Cai ら [26])**.** 任意のパラメータ化問題 L において，L が *FPT* であることと L にカーネル化が存在することは同値．

よってパラメータ問題が FPT であることを示すにはカーネル化の存在を示せば十分である．それだけでなく，カーネル化はパラメータの小さい問題に対して効率のよいデータ前処理の方法であり，その応用は多岐にわたる．

1.3.2　厳密指数時間アルゴリズム

前小節では，パラメータが小さいときに高速に動作する厳密アルゴリズムの考え方を紹介した．本小節では，厳密指数時間アルゴリズム [113, 50] に関して導入する．

NP 困難問題に対する最もシンプルな解法は**しらみ潰し法**である．NP に属する問題は，解の候補をすべて列挙することができれば，その多項式倍の時間で計算可能である．例えば，n 要素集合が与えられたとき，ある性質をもつ部分集合を求める問題は，その部分集合が性質を持つかどうかを多項式時間で判定できれば $2^n n^{O(1)}$ 時間で解くことができ，部分集合の代わりに順列や分割を求める問題である場合は，それぞれ $n!n^{O(1)}$ 時間，$n^n n^{O(1)}$ 時間で解くことができる．

さらに大きいインスタンスの問題を解くために，多くの指数時間アルゴリズムの研究が行われている．最も有名な例として Bellman[9] および Held-Karp[62] の巡回セールスマン問題の結果を挙げる．巡回セールスマン問題は，n 都市からなる集合とその都市間の距離が与えられたとき，ある都市から始めて，すべての都市をちょうど 1 回ずつ訪問して最初の都市に戻ってくるときの総移動距離を最小化する問題である．この問題は都市集合上の順列を求める問題なので，$n!n^{O(1)}$ 時間のしらみ潰し法アルゴリズムが存在するが，Bellman，Held-Karp は $2^n n^2$ 時間の動的計画法を与えた．現在のコンピュータを用いると，しらみ潰し法アルゴリズムに比べてこの動的計画法は 2 倍以上の大きさのインスタンスを解くことができる．その一方で，５０年以上の研究が行われたにもかかわらず，巡回セールスマン問題に対する $2^n n^2$ 時間アルゴリズムを改善する結果は知られていない．

また，指数時間アルゴリズムの研究が行われているもう一つの理由は，NP 困難問題に対する組合せ論的かつアルゴリズム論的理解のためである．グラフの独立集合とは頂点集合の部分集合で，どの辺の両端点も同時にその部分集合に含まれないようなものである．最大独立集合問題は，n 頂点グラフが与えられて要素数最大の独立集合を求める問題である．この問題

はしらみ潰し法で $2^n n^{O(1)}$ 時間で解くことができるが，以下の Moon-Moser の古い結果 [87] を用いることで計算時間を改善できる．

定理 1.3 ([87])**.** n 頂点グラフの極大な独立集合の数は $O(3^{n/3})$ である．

この定理を用いることで $3^{n/3}n^{O(1)}$ 時間アルゴリズムが設計できる．このような古くから知られる組合せ論的な解析結果を指数時間アルゴリズムに応用することは多くなされており，組合せ論と指数時間アルゴリズムそれぞれの理解が，もう一方へ新しい視点を与えることができる．

1.4　本論文の構成

本学位請求論文の結果は以下の出版された論文からなる：

- [80] Yasuaki Kobayashi and Hisao Tamaki, A fast and simple subexponential fixed parameter algorithm for one-sided crossing minimization. In *Proceedings of the 20th Annual European Symposium on Algorithms*, volume 7501 of *Lecture Notes in Computer Science, ESA'12*, pages 683–694, 2012.
- [79] Yasuaki Kobayashi, Hirokazu Maruta, Yusuke Nakae and Hisao Tamaki, A linear edge kernel for two-layer crossing minimization. In *Proceedings of the 19th Annual International Computing and Combinatorics Conference*, volume 7936 of *Lecture Notes in Computer Science, COCOON'12*, pages 459–468, 2013.
- [75] Kenta Kitsunai, Yasuaki Kobayashi, Keita Komuro, Toshihiro Tano and Hisao Tamaki, Computing directed pathwidth in $O(1.89^n)$ time. In *Proceedings of the 7th International Symposium on Parameterized and Exact Computation*, volume 7535 of *Lecture Notes in Computer Science, IPEC'12*, pages 182–193, 2012.

最初の論文の結果は 2 章にまとめる．また，[80] では OSCM を解く $O(3^{\sqrt{2k}} + n)$ 時間アルゴリズムを与えたが，本論文では $O(k2^{\sqrt{2k}} + n)$ に改善している．3 章では，[79] の結果をまとめる．同様の結果は中江の学位論文 [115] にあるが，カーネルの辺数とカーネル化の実行時間を改善している．詳細は 3 章．最後の結果 [75] は 4 章にまとめる．また，2 章，3 章，4 章の間には依存関係はなく，独立に読むことができる．

第2章　One-sided crossing minimization

本章では，One-Sided Crossing Minimization(OSCM) を解く準指数時間固定パラメータアルゴリズムを与える．以下の問題を考える．

> ONE-SIDED CROSSING MINIMIZATION(OSCM)
> **インスタンス**：n 頂点 2 部グラフ $G = (X(G), Y(G), E(G))$，$X(G)$ 上の全順序 $<_X$ と正整数 k
> **タスク**：$Y(G)$ 上の全順序 $<_Y$ で $\mathrm{bcr}(\mathcal{D} = (G, <_X, <_Y)) \leq k$ であるようなものが存在するか？

定理 2.1. *OSCM* を解く $O(k2^{\sqrt{2k}} + n)$ 時間アルゴリズムが存在する．

本結果の会議版論文 [80] では，OSCM を解く $O(3^{\sqrt{2k}} + n)$ 時間アルゴリズムを与えた．本章では，定理 2.1 の実行時間に改善している．

さらに，実行時間の指数部 $O(\sqrt{k})$ は指数時間仮説 [66] が成り立つとすると漸近的に最適であることを示す．指数時間仮説とは，各 $t \geq 3$ において，ある定数 c_t が存在し，t-SAT が $O(2^{c_t n})$ 時間で解くことができないという仮説である．ここで，n は変数の数とする．

定理 2.2. 指数時間仮説が成り立つならば，*OSCM* を解く $2^{o(\sqrt{k})}n^{O(1)}$ 時間アルゴリズムは存在しない．

2.1 節では，OSCM に関する先行研究とその応用について述べ，2.2 節では，定理 2.1 を示す．2.3 節では，定理 2.2 を示す．2.4 節では，本結果のまとめと今後の展望に関して述べる．

2.1　研究背景

OSCM は有向グラフの多層描画へのアプローチとして有名な “Sugiyama method” [108] の重要な部分問題として位置づけられている．*daVinci* [53], *GraphViz* [55] 内のツールである *dot* [54]，*GraphLet* [63] などのシステムがこの方法に基づいてグラフの描画を行っている．Sugiyama method は以下のステップで構成される．

1. 入力された有向グラフの辺の向きを反転させ，有向非閉路グラフに変形する．これは NP 困難として知られる有向フィードバック辺集合問題 [56] と一致する．(有向フィードバック辺集合問題も 1-グラフレイアウト問題である．)

2. 頂点集合を幾つかの "層" $L_1, L_2, \ldots, L_h$ に割り当てる．このとき，各辺は上層から下層へ向かうようにする．また，ある割り当てにおいて，辺 (u, v) の**スパン**とは u が割り当てられている層 L_i と v が割り当てられている層 L_j としたとき，$j - i$ と定義する．このステップではスパンが 1 より大きい辺ができるだけ少なくかつ各層への頂点数がバランスするように頂点を層に割り当てる問題を解く．割り当てが決定したあと，スパンが 1 より大きい辺に関してはダミー頂点を挿入することでスパンを 1 に減らす.

3. 各層に割り当てられて頂点集合を並び替えて，交差数を最小化する．このステップの詳細は後述する．

4. 相対的な順序が決められた頂点集合に対して，それぞれの座標を決定する．

3 番目のステップは以下のように行う．有向グラフ $G = (V(G), E(G))$ とその頂点集合から層 $L_1, L_2, \ldots, L_h$ への割り当てが与えられたとき，$i = 1, 2, \ldots, h - 1$ の順番に L_i と L_{i+1} の 2 層から成る 2 部グラフにおいて OSCM を解く．このとき，L_i の層の頂点順序を固定することで L_{i-1} と L_i の 2 層で解いた OSCM の最適交差数は保存される．図 2.1 はその一例である．

これにより Sugiyama method の中で OSCM は重要な 1 ステップを担う．

OSCM は NP 完全であることが知られており [45]，グラフが疎な場合でも NP 完全である [89]．[89] の NP 完全性の結果では，インスタンス $(G, <_X, k)$ の $Y(G)$ に含まれる頂点の次数は 4 であるような森でも NP 完全であることを示した一方で，$Y(G)$ に含まれる頂点の次数 2 のときの多項式時間アルゴリズムを与えている．$Y(G)$ に含まれる頂点の次数 3 のときは未解決問題である．Dujmović-Whitesides[41] は OSCM が FPT であることを，$O(\phi^k n^2)$ 時間固定パラメータアルゴリズムを与えること

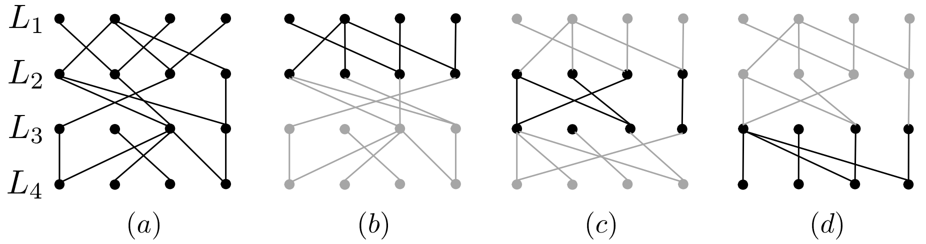

図 2.1: $(a)\{L_1, L_2, L_3, L_4\}$ の 4 層からなるグラフ．$(b)L_1$ 上の頂点順序を固定して L_1 上の頂点と L_2 上の頂点で OSCM を解く．$(c)L_2$ 上の頂点順序を固定して L_2 上の頂点と L_3 上の頂点で OSCM を解く．$(d)L_3$ 上の頂点順序を固定して L_3 上の頂点と L_4 上の頂点で OSCM を解く．

で示した．ここで，$\phi \approx 1.6182$ は黄金比である．この結果は Dujmović-Fernau-Kaufmann[39] によって $O(1.4656^k + kn^2)$ に改善された．さらに，Fermau ら [47] は OSCM を重み付きトーナメント上のフィードバック辺集合問題 (FAST) に帰着し，Alon-Lokshtanov-Saurabh の FAST の準指数時間固定パラメータアルゴリズム [2] を利用することで，OSCM に対する $2^{O(\sqrt{k}\log k)} + n^{O(1)}$ 時間アルゴリズムを与えた．この帰着は，[72] の FAST に対する多項式時間近似スキームの結果を利用することで，OSCM の多項式時間近似スキームへも利用できる．さらに，OSCM は標準的な動的計画法を用いて $O(2^{|Y(G)|}n)$ 時間で解くことができる [19].

実際のアプリケーションでは OSCM はヒューリスティックアルゴリズムを用いて解くことが一般的である．Eades-Kelly[42] は *greedy insertion*, *greedy switch*，*split* と呼ばれるアルゴリズムを 1986 年に与えた．Eades-Wormald[45] は *median heuristic* と呼ばれるアルゴリズムを与え，そのアルゴリズムの近似率が 3 であることを示した．Nagamochi[90] は median heuristic をベースにランダム化を用いて OSCM の 1.4664 近似アルゴリズムを与えた．さらに Nagamochi は密なグラフに対してさらに良い近似率のアルゴリズム [91] を与えた．他にも，*bary center* [108]，*assignment heuristic* [27]，*stochastic heuristic* [36] などが知られている．

本結果の固定パラメータアルゴリズムは，既存の固定パラメータアルゴリズムに比べて高速である．特に，実行時間において指数的に依存する k の関数と多項式的に依存する n の関数を同時に改善している．さらに詳細には，[47] の結果と比較すると，本結果の実行時間の指数部には隠

れた定数はない．これらにより，本結果のアルゴリズムが理論的に高速なだけでなく，実用上の観点からも [47] と比べて高速に動作することが期待される．また，特筆すべきは，実行時間の k の指数関数と n の多項式関数をカーネル化を用いずに分離している．

本結果のもう一つの優位点は，アルゴリズムのシンプルさである．[47] のアルゴリズムは，重み付きトーナメントのフィードバック辺集合問題に帰着し，[2] のカーネル化や choromatic coding といったテクニック，および決定性アルゴリズムを得るための universal coloring family を用いた脱乱択化など必要であるが，本結果のアルゴリズムは標準的な動的計画法のみであり，[47] のアルゴリズムと比較すると，大幅に単純化されている．

本結果のアルゴリズムは，[41, 39] の結果の延長線上にある．[41, 39] の結果では，2.2 節の述べる区間表現を暗黙的に利用し，有界探索木のアルゴリズムを設計している．本結果では，その区間表現を区間グラフと見なし，区間グラフのパス分解を用いた動的計画法を行う．

2.2　準指数時間固定パラメータアルゴリズム

本節では，定理 2.1 の証明を行う．

2.2.1　諸定義

本章で用いる記法や用語の定義を行う．本章では，グラフ G は無向 2 部グラフであるとし，頂点数 $|V(G)|$ を n で，辺数 $|E(G)|$ を m で表す．$X(G)$ の頂点集合は $<_X$ の順序で並べられたリストとして与えられているとする．

任意の異なる 2 頂点 $u, v \in Y(G)$ において，

$$c(u,v) = |\{(x,x') : x \in N(u), x' \in N(v), x' <_X x\}|$$

と定義する．(文献によっては，c_{uv} と記述することがあるが，以下で集合上へ拡張のためにこのような記法を用いる．) これは，Y 層において $u <_Y v$ としたときの，u と v のいずれかに接続される辺集合に生じる辺交差数と一致する．また，この表記法は以下のように集合上へ拡張できる：互いに素な $Y(G)$ の部分集合 U, V において，$c(U,V) = \sum_{u \in U, v \in V} c(u,v)$ とする．$U \subseteq Y(G)$ と U 上の順列 $\pi \in \Pi(U)$ において，$c(\pi)$ を，U を π

によって並べた時の，U に接続される辺集合に生じる交差数とする．つまり，

$$c(\pi) = \sum_{u,v \in U, u <_\pi v} c(u,v)$$

とする．各 $U \subseteq Y(G)$ において，$\mathrm{opt}(U) = \min\{c(\pi) : \pi \in \Pi(U)\}$ とする．このとき，OSCM の目的は $\mathrm{opt}(Y(G)) \leq k$ であるかを判定することである．$\mathrm{opt}(U) = c(\pi)$ を満たす U の順列 π を**最適な順列**と呼ぶ．

以下の補題は Dujmović-Whitesides[41] によるものである．

補題 2.1 (Dujmović-Whitesides[41])**.** u と v を $Y(G)$ の異なる頂点とし，$c(u,v) = 0$ かつ $c(v,u) > 0$ であるとする．このとき，任意の最適な $Y(G)$ の順列 π において $u <_\pi v$ を満たす．

$Y(G)$ の異なる 2 点からなる非順序対 $\{u,v\}$ において，$c(u,v) = 0$ かつ $c(v,u) > 0$ のとき，$\{u,v\}$ は (u,v) に**順序付けられた**といい，また，$\{u,v\}$ は**順序付けられた対**と呼ぶ．さらに，$c(u,v) > 0$ かつ $c(v,u) > 0$ のとき $\{u,v\}$ を**順序付け可能対**であるといい，$c(u,v) = c(v,u) = 0$ のとき，$\{u,v\}$ は**自由対**であるという．これらを用いて，以下の系が成り立つ．

系 2.1. π を $Y(G)$ の最適な順列とし，u, v を $Y(G)$ の異なる 2 点とする．このとき，$\{u,v\}$ が (u,v) に順序付けられているならば，$u <_\pi v$ であり，$\{u,v\}$ が自由対の場合，π において u と v の位置を交換して得られる順列 π' も最適な順列である．

証明. $\{u,v\}$ が (u,v) に順序付けられているならば，補題 2.1 より，$u <_\pi v$ である．また，$\{u,v\}$ が自由対であることは，ある $x \in X(G)$ を用いて，$N(u) = N(v) = \{x\}$ であることと等価なので，$c(\pi) = c(\pi')$ より，系の後半部分も成り立つ．　$\square$

各順序付け可能対は少なくともひとつの交差を生じさせることから，以下が成り立つ．

命題 2.1. $\mathrm{opt}(Y(G)) \leq k$ ならば，順序付け可能対の個数は高々 k である．

[41, 39] のアルゴリズムは，補題 2.1 を用いて有界探索木のアルゴリズムを設計している．

2.2.2　区間表現

本小節では，OSCM インスタンス $(G, <_X, k)$ から得られる 2 つの異なる区間表現を説明する．異なる区間表現を定義する理由は，続く 2 つの小節において，異なる使われ方をすることに起因する．また，本章の以降では G が孤立点，つまり次数 0 の頂点，を持たないと仮定する．

素朴な区間表現: 素朴な区間表現においては，$Y(G)$ の頂点はすべて次数が 2 以上であると仮定する．(この仮定の詳細は次の小節 2.2.3 で説明する.) $y \in Y(G)$ において，$l_y(r_y)$ を $N(y)$ のなかで，$<_X$ に関して最小 (最大) の $x \in N(y)$ と定義する．半区間 $I_y = [l_y, r_y) = \{x \in X(G) : l_y \leq_x x <_x r_y\}$ とし，半区間の多重集合 $\mathcal{I} = \{I_y : y \in Y(G)\}$ とする．上記の仮定より，各半区間は空でない．これらの区間集合は $O(n+m)$ 時間で計算可能である．

この区間表現において以下の観察ができる．

観察 2.1. 異なる $u, v \in Y(G)$ において，$I_u \cap I_v \neq \emptyset$ であることと，$c(u,v) > 0$ かつ $c(v,u) > 0$ であることが同値．

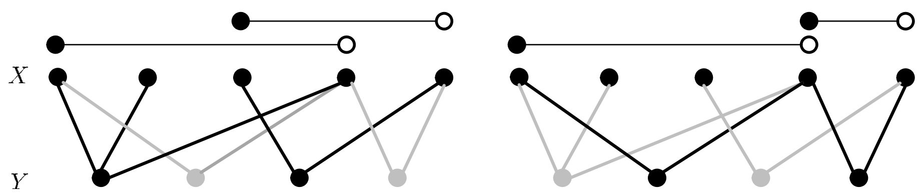

図 2.2: 素朴な区間表現の例．異なる $u, v \in Y(G)$ において，$I_u \cap I_v \neq \emptyset$ と $c(u,v) > 0$ かつ $c(v,u) > 0$ であることが同値である．

よって，$\{u, v\}$ が順序付け可能対であるならば，$I_u \cap I_v \neq \emptyset$ である．さらに $Y(G)$ の相異なる t 個の頂点 $u_1, u_2, \ldots, u_t$ において，$I_{u_1} \cap I_{u_2} \cap \ldots \cap I_{u_t} \neq \emptyset$ ならば，そのインスタンスが少なくとも $\frac{t(t-1)}{2}$ 個の辺交差を持つことがわかる．よって以下が導かれる．

補題 2.2. *OSCM の YES インスタンス* $(G, <_X, k)$ とそのインスタンスから得られる素朴な区間表現 $\mathcal{I}$ において，相異なる t 個の頂点 $u_1, u_2, \ldots, u_t \in Y(G)$ が $I_{u_1} \cap I_{u_2} \cap \ldots \cap I_{u_t} \neq \emptyset$ を満たすとき，$t \leq \sqrt{2k} + 1$ である．

この事実は，小節 2.2.4 で用いる．

摂動された区間表現: 素朴な区間表現 $\mathcal{I}$ とは異なる区間表現 $\mathcal{J} = \{J_y : y \in Y(G)\}$ を定義する．この区間表現においては，次数の制約は仮定しない．$y \in Y(G)$ において，l_y, r_y を素朴な区間表現のときと同様に定義する．次数の制約を仮定していないので，$l_y = r_y$ となりうることに注意する．ここで，各 $y \in Y(G)$ において，$J_y = [a_y, b_y]$ を以下の条件を満たすように定義する．

J1 $1 \leq a_y < b_y \leq 2|Y(G)|$,

J2 各 $1 \leq t \leq 2|Y(G)|$ において，$a_y = t$ または $b_y = t$ のいずれか一方を満たす $y \in Y(G)$ が唯一存在,

J3 $u, v \in Y(G)$ において $b_u < a_v$ ならば $c(u, v) = 0$.

条件 J1，J2 は小節 2.2.4 の動的計画法のため，J3 は観察 2.1 を満たすためである．また $\mathcal{J}$ においても補題 2.2 と同等の結論を導くことができる．以下では，OSCM のインスタンス $(G, <_X, k)$ から $\mathcal{J}$ を $O(n+m)$ 時間で得る方法を示す.

$P = \{(y, l_y, \mathrm{left}) : y \in Y(G)\} \cup \{(y, r_y, \mathrm{right}) : y \in Y(G)\}$ とする．これらは，各 $y \in Y(G)$ と対応する区間 J_y の左端と右端を表現している．$\mathcal{J}$ を得る方法は，P 上の全順序を定義し，その順序における $(y, l_y, \mathrm{left}), (y, r_y, \mathrm{right})$ のランクによって a_y, b_y の値を決定する．各 $p \in P$ において，$y(p), x(p), e(p)$ それぞれを p の第１，２，３要素とする．P 上の全順序関係 $<$ を以下のように定義する．

全順序関係 $<$ は基本的には $<_X$ に従う：$p, q \in P$ において $x(p) <_X x(q)$ ならば $p < q$ である．各 $x \in X(G)$ において，$P_x = \{p \in P : x(p) = x\}$ とする．$P_x^1 = \{p \in P_x : d(y(p)) > 1, e(p) = \mathrm{right}\}$，$P_x^2 = \{p \in P_x : d(y(p)) = 1\}$，$P_x^3 = \{p \in P_x : d(y(p)) > 1, e(p) = \mathrm{left}\}$ とする．ここで，$i < j$ を満たす $p \in P_x^i$ かつ $q \in P_x^j$ であるならば，$p < q$ と定義する．さらに，P_x^1，P_x^3 内の順序は任意に決め，P_x^2 においては，同一の $y \in Y(G)$ を含む対においては $(y, x, \mathrm{left}) < (y, x, \mathrm{right})$ とし，これら対同士が “交差しない” ように (異なる $y, y' \in Y(G)$ において，$(y, x, \mathrm{left}) < (y', x, \mathrm{left}) < (y, x, \mathrm{right}) < (y', x, \mathrm{right})$，または $(y, x, \mathrm{left}) < (y', x, \mathrm{left}) < (y', x, \mathrm{right}) < (y, x, \mathrm{right})$ であるような y, y' が存在しないように) 順序関係を決定する．

よって，P 上の全順序関係 $<$ が定義され，そこから得られる順列 $\pi = p_1, p_2, \ldots, p_{|2Y(G)|}$ ($i < j$ ならば $p_i < p_j$) としたとき，$p_t = (y, l_y, 0)$ ならば $a_y = t$，$p_t = (y, r_y, 1)$ ならば $b_y = t$ とする．これより，$\mathcal{J}$ を構成することができる．

P を構成するのは，$O(n+m)$ 時間で可能であり，P を P_x へ分割するのは $O(|P|)$ 時間，P_x を P_x^1, P_x^2, P_x^3 へ分割，および P_x^2 の順序関係を決定するのは，それぞれ $O(|P_x|)$ 時間，$O(|P_x^2|)$ 時間で可能である．よって，$\mathcal{J}$ の構成は $O(n+m)$ 時間で可能である．

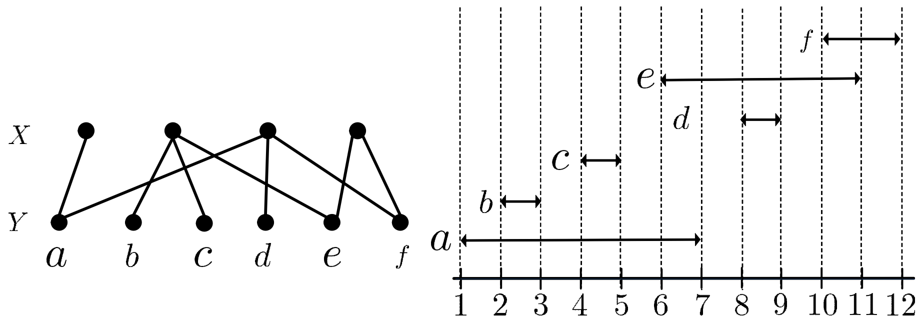

図 2.3: 摂動された区間表現の例．条件 J1，J2，J3 を満たしている．

命題 2.2. 区間表現 $\mathcal{J}$ は条件 *J1*，*J2*，*J3* を満たす．

証明. $\mathcal{J}$ は P 上の全順序から得ているので，明らかに条件 J1，J2 を満たす．さらに，$b_u < a_v$ ならば $r_u \leq_X l_v$ なので J3 も満たす．　□

系 2.1 は以下のように摂動された区間表現 $\mathcal{J}$ を用いて言い直すことができる．$U \subseteq Y(G)$ の順列 π が $\mathcal{J}$ と**整合している**とは，任意の 2 点 $u, v \in U$ において $b_u < a_v$ ならば $u <_\pi v$ を満たすことである．

補題 2.3. $U \subseteq Y(G)$ とする．このとき，$\mathcal{J}$ と整合している最適な順列 $\pi \in \Pi(U)$ が存在する．

証明. 各 $x \in X(G)$ において，$U_x = \{y \in U : N(y) = \{x\}\}$ とする．各 x において U_x の異なる 2 点は自由対なので，系 2.1 を U_x へ適用することにより，U_x は $\mathcal{J}$ に整合していると仮定できる．ここで，$u, v \in U$ において $b_u < a_v$ と仮定する．$\mathcal{J}$ の条件 J3 より，$c(u, v) = 0$ である．$c(v, u) > 0$ ならば，系 2.1 より $\{u, v\}$ は (u, v) に順序付けられるので，$u <_\pi v$ であり，$c(v, u) = 0$ ならば，$\{u, v\}$ は自由対なので，ある $x \in X(G)$ において $u, v \in U_x$ を満たすので π は $\mathcal{J}$ に整合している．　□

2.2.3　交差数を計算する線形時間アルゴリズム

Dujmović-Whitesides のアルゴリズム [41] では，すべての異なる $u, v \in Y(G)$ の対において $c(u, v)$ を $O(kn^2)$ 時間で計算している．本小節では，すべての順序付け可能対 $\{u, v\}$ において $c(u, v)$ と $c(v, u)$ を $O(n + k)$ 時間で計算するか，与えられたインスタンスが NO インスタンスであることを出力する．

ここでは，素朴な区間表現 $\mathcal{I} = \{I_y = [l_y, r_y) : y \in Y(G)\}$ を用いる．また，この小節では $<_X$ の代わりに $<$，$\leq_X$ の代わりに $\leq$ を用いる．

各 $y \in Y(G)$ と $x \in X(G)$ において，$d^{<x}(y) = |\{z \in N(y) : z < x\}|$, $d^{\leq x}(y) = |\{z \in N(y) : z \leq x\}|$ とする．

以下の等式を用いることにより，$c(u, v)$ を計算する．

$$c(u, v) = \left[\sum_{x \in N(u), l_v < x \leq r_v} d^{<x}(v) \right] + d(v) \cdot (d(u) - d^{\leq r_v}(u)). \tag{2.1}$$

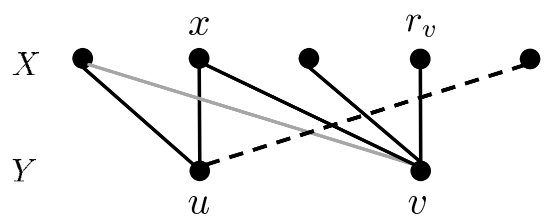

図 2.4: $d^{<x}$(v) によって灰色辺と辺 $\{x, u\}$ の交差がカウントされ，$d(v) - d^{\leq r_v}(u)$ 本の辺 (点線) は v に接続するすべての辺と交差する．

各 $x \in X(G)$ において，$Y_x = \{y \in Y(G) : l_y < x < r_y\}$ とする．本小節においては，非順序対 $\{u, v\}$ が順序付け可能対であるとき，対応する順序対 $(u, v), (v, u)$ が**順序付け可能対**であるという．等式 (2.1) を用いて，$c(u, v)$ をカウンタ $c[u, v]$ を用いてカウントする．このカウンタには，$|Y| \times |Y|$ の 2 次元配列を用いて，(u, v) が順序付け可能対であるときのみ $c[u, v]$ を 0 で初期化する．交差数計算のアルゴリズムは以下の 2 ステップからなる．

1. Y_x を更新しながら $<$ の順序で $X(G)$ の頂点を走査する．ある点 $x \in X(G)$ において，各 $u \in N(x)$ と各 $v \in Y_x$ について $c[u, v]$ を 0 で初期化する．

2. 同様に Y_x と各 $y \in Y(G)$ において $d^{<x}(y)$ を更新しながら　$<$ の順序で $X(G)$ の頂点を走査する．ある点 $x \in X(G)$ に到達したとする．各 $u \in N(x)$ と $v \in Y_x$ において $d^{<x}(v)$(等式 (2.1) の第 1 項) を $c[u,v]$ に加える．さらに，$r_v = x$ なる各 $u \in Y_x$ と $v \in N(x)$ において，$d(v) \cdot (d(u) - d^{\leq x}(u))$(等式 (2.1) の第 2 項) を $c[u,v]$ に加える．

補題 2.4. *OSCM* のインスタンスが *YES* インスタンスならば，上記のアルゴリズムの実行時間は $O(n+k)$ である．

証明. インスタンスが OSCM の YES インスタンスであると仮定する．上記のアルゴリズムにおいて，Y_x を双方向リストで記憶する．これにより，頂点の追加と削除を $O(1)$ 時間で達成する．各 $y \in Y(G) x \in X(G)$ を走査する過程で，$l_y = x$ となった直後に Y_x に追加され，$r_y = x$ を満たしたとき Y_x から取り除かれる．よってアルゴリズム全体では，Y_x は $O(n+m)$ 時間で更新可能である．ステップ 1 では，$u \in N(x)$ と $v \in Y_x$ なる対に対してのみ $c[u,v]$ の初期化を行う．このとき，$I_u \cap I_v \neq \emptyset$ であるので，命題 2.1 と観察 2.1 より，高々$2k$ 回の初期化しか起こらない．よって，ステップ 1 の実行時間は $O(n+m+k)$ である．ステップ 2 では，各 $v \in Y(G)$ において $d^{<x}(v)$ の値を管理する必要があるが，その値をカウントするカウンタを用意し，$x \in X(G)$ に対する処理が行われた後で，$v \in N(x)$ に対して 1 だけインクリメントすれば更新可能である．よってステップ 2 も $O(n+m)$ 時間で実行可能である．補題 1.2 より，アルゴリズム全体は $O(n+k)$ 時間で実行できる．□

任意のインスタンスで目的の時間を達成するためには，補題 1.2 の条件を満たすことと，ステップ 1 において何回の初期化が起こったかをカウントする必要がある．条件が満たされていないか，初期化の回数が $2k$ を超えるならば，アルゴリズムを止めて，NO インスタンスであることを出力すればよい．

また，上記のアルゴリズムでは $O(|Y|^2)$ 空間必要であるが，2 分探索木でカウンタを管理することにより，$O(n+k)$ 空間，$O((n+k)\log k)$ 時間で計算することも可能である．

2.2.4　区間表現上の動的計画法

本小節では，$\mathrm{opt}(Y(G))$ を求める動的計画法を説明する．前の小節より，すべての順序付け可能な対 $\{u,v\}$ において $c(u,v), c(v,u)$ が計算済み

であることを仮定する．また，本小節では摂動された区間表現 $\mathcal{J} = \{J_y = [a_y, b_y] : y \in Y(G)\}$ を用いる．

各 $t1 \leq t \leq 2|Y(G)|$ において，$L_t = \{y \in Y(G) : b_y \leq t\}$，$M_t = \{y \in Y(G) : a_y \leq t < b_y\}$，$R_t = \{y \in Y(G) : t < a_y\}$ とする．ここで $t > 1$ のとき，

A. ある $y \in Y(G)$ において $t = a_y$ ならば，$L_t = L_{t-1}, M_t = M_{t-1} \cup \{y\}, R_t = R_{t-1} \setminus \{y\}$ であり，

B. ある $y \in Y(G)$ において $t = b_y$ ならば，$L_t = L_{t-1} \cup \{y\}, M_t = M_{t-1} \setminus \{y\}, R_t = R_{t-1}$ である．

各 $1 \leq t \leq 2|Y(G)|$ において，以下の 2 つの値を計算し，表に記憶する：(1) 各 $y \in M_t$ について $c(L_t, \{y\})$，(2) 各 $S \subseteq M_t$ について $\mathrm{opt}(L_t \cup S)$.

(1) の再帰式は次のようになる．$t = 1$ に関して，$c(L_1, \{y\}) = 0$ である．ここで，$L_1 = \emptyset$，y は M_t の唯一の要素である．$2 \leq t \leq 2|Y(G)|$ においては，A の条件を満たせば，$L_t = L_{t-1}$ より, 各 $v \in M_t \setminus \{y\}$ について $c(L_t, \{v\}) = c(L_{t-1}, \{v\})$ である．また，各 $u \in L_t$ について $b_u < a_y$ なので，$c(L_t, \{y\}) = 0$ である．次に，B の条件を満たすときを考える．各 $v \in M_t$ において，$c(L_t, \{v\}) = c(L_{t-1} \cup \{y\}, \{v\}) = c(L_{t-1}, \{v\}) + c(y, v)$ である．さらに，$\{y, v\}$ は順序付け可能対であり，$c(y, v)$ はすでに計算済みであるので，$c(L_{t-1}, \{v\})$ は計算可能である．どちらの場合も各 $1 \leq t \leq 2|Y(G)|$ について，$h = |M_t|$ とすると，$O(h)$ 時間で (1) は計算可能である．

次に，(2) の再帰式を計算する．以下の補題は (2) の再帰式を計算する上で重要である．

補題 2.5. $L_{t-1} \cup S$ の最適な順列 π で，最後の頂点が S に含まれるものが存在する．

証明. 補題 2.3 より，π は $\mathcal{J}$ と整合している．π が $\mathcal{J}$ に整合していることと，任意の $u \in L_t$ において $b_u < a_y$ であることより，$u <_\pi y$ である．よって，ある $v \in S$ が存在し，$y <_\pi v$ または $y = v$ のいずれかを満たす．　□

補題 2.6. t を $1 \leq t \leq 2|Y(G)|$ を満たす整数とし，$h = |M_t|$ とする．すべての $v \in M_t$ について，$c(L_t, \{v\})$ の値と，すべての $S \subseteq M_{t-1}$ について，$\mathrm{opt}(L_{t-1} \cup S)$ の値が表で与えられているとすると，すべての $S \subseteq M_t$ について，$\mathrm{opt}(L_t \cup S)$ を $O(h2^h)$ 時間で計算可能である．

証明. まず，各 $v \in M_t$ と $S \subseteq M_t \setminus \{v\}$ において，$c(S, \{v\})$ を計算し，表に記憶する．これは $O(h2^h)$ 時間で可能である．

B の条件 (ある $y \in Y(G)$ において $t = b_y$) を満たすと仮定する．ここで，$L_t = L_{t-1} \cup \{y\}$ である．任意の $S \subseteq M_t$ において $(S \cup \{y\}) \subseteq M_{t-1}$ なので，$\mathrm{opt}(L_t \cup S) = \mathrm{opt}(L_{t-1} \cup (S \cup \{y\}))$ は既に表に計算済みである．

次に A の条件 (ある $y \in Y(G)$ において $t = a_y$) を満たすと仮定する．ここで，$L_t = L_{t-1}$ であり，$M_t = M_{t-1} \cup \{y\}$ である．$S \subseteq M_t$ する．もし，$y \notin S$ ならば，$\mathrm{opt}(L_t \cup S) = \mathrm{opt}(L_{t-1} \cup S)$ なので，表に計算済みである．また，$y \in S$ ならば，補題 2.5 より，$L_t \cup S$ のある最適な順列 π が存在して，π の最後の頂点は S に属している．よって，

$$
\begin{aligned}
\mathrm{opt}(L_t \cup S) &= \min_{v \in S}\{\mathrm{opt}(L_t \cup S \setminus \{v\}) + c(L_t \cup S \setminus \{v\}, \{v\})\} \\
&= \min_{v \in S}\{\mathrm{opt}(L_t \cup S \setminus \{v\}) + c(L_t, \{v\}) + c(S \setminus \{v\}, \{v\})\}
\end{aligned}
$$

である．第 2 項と 3 項は表に計算済みであるので，動的計画法を用いて，すべての $S \subseteq M_t$ について $\mathrm{opt}(L_t \cup S)$ を $O(h2^h)$ 時間で計算可能である． □

$h = |M_t|$ とすると，補題 2.2 を $\mathcal{J}$ に適用すると $h \leq \sqrt{2k} + 1$ を得る．この上界と，$1 \leq t \leq 2|Y(G)|$ において $|M_t| \geq 2$ を満たす異なる t は高々 k 通りなので，$H = \lceil\sqrt{2k}\rceil + 1$ とするとアルゴリズムの実行時間は

$$
\begin{aligned}
\sum_{1 \leq t \leq 2|Y(G)|} O(|M_t| 2^{|M_t|}) &= O(\sum_{2 \leq h \leq H} \sum_{t : |M_t| = h} h2^h + n) \\
&= O(k \cdot \sum_{2 \leq h \leq H} h2^h + n) \\
&= O(k^{3/2} 2^{\sqrt{2k}} + n).
\end{aligned}
$$

さらなるタイトな解析のために以下の補題を示す.

補題 2.7. $\mathcal{J}$ が高々 k の互いに交わる区間の対を含んでいるとする．$H = \lceil\sqrt{2k}\rceil + 1$ とし，$2 \leq h \leq H$ において c_h を $|M_t| = h$ となるような，異なる t の通り数とする．このとき，$c_h \leq 2^{H-h+2}$ である．

証明. $2 \leq h \leq H$ なる h を固定する．$t_1 < t_2 < \ldots < t_{c_h}$ を $\{t : |M_t| = h\}$ の各要素とする．M_{t_1} は $\frac{h(h-1)}{2}$ 個の順序付け可能対を含んでいる．ここで，$2 \leq i \leq c_h$ において，M_{t_i} は少なくとも $h - 1$ 個の新たな順序付け可能対を生じさせることを示す．$M_{t_i} \neq M_{t_{i-1}}$ の場合，ある $u \in M_{t_i} \setminus M_{t_{i-1}}$

が存在し，各 $v \in M_{t_i} \setminus \{u\}$ について $\{u, v\}$ が新たな順序付け可能対となる．$M_{t_i} = M_{t_{i-1}}$ の場合，$M_{t_{i-1}} \subset M_{t_{i-1}+1}$ なので，$u \in M_{t_{i-1}+1} \setminus M_{t_{i-1}}$ が存在し，各 $v \in M_{t_i} \setminus \{u\}$ について $\{u, v\}$ が新たな順序付け可能対となる．よって，$\frac{h(h-1)}{2} + (c_h - 1)(h-1) \leq k$ である．この不等式を解くと，$c_h \leq \frac{k}{h-1} - \frac{h}{2} + 1$ が得られる．

$H - h + 2 \geq \log_2 k$ の場合，$c_h \leq 2^{H-h+2}$ は明らかである．よって，$j = H - h + 2 < \log_2 k$ とする．$H \geq \sqrt{2k} + 1 \geq 2(\log_2 k - 1)$ より，$(H - j + 1)(H + 2j) = H^2 + H + j(H - 2j + 2) > H^2 > 2k$ である．よって，$\frac{k}{h-1} = \frac{k}{H-j+1} < \frac{H+2j}{2} = \frac{H}{2} + j$ を満たし，

$$c_h \leq \frac{H}{2} + j - \frac{H - j + 2}{2} + 1 = \frac{3j}{2} \leq 2^j. \tag{2.2}$$

よって，補題が得られる．□

補題 2.8. *OSCM インスタンスが YES インスタンスと仮定する．このとき，補題 2.6 の動的計画法は* $O(k2^{\sqrt{2k}} + n)$ *時間で実行できる．*

証明. H と c_h $(2 \leq h \leq H)$ を補題 2.7 と同様に定義する．補題 2.6 より，動的計画法の実行時間は $O((\sum_{2 \leq h \leq H} c_h h^2 h) + n)$ である．補題 2.7 より，

$$\begin{aligned}
\sum_{2 \leq h \leq H} c_h h 2^h &\leq \sum_{2 \leq h \leq H} 2^{H-h+2} h 2^h \\
&= \sum_{2 \leq h \leq H} h 2^{H+2} \\
&\leq 2^{H+1} H(H+1) \\
&= O(k 2^{\sqrt{2k}})
\end{aligned}$$

よって，補題が得られる．□

2.2.5　アルゴリズムの全体像

アルゴリズムの全体像を Algorithm 1 にまとめる．$\mathrm{opt}(\cdot)$ や $c(\cdot,\cdot)$ はプログラムの変数として用いていることに注意．小節 2.2.2，2.2.3，2.2.4 より，実行時間は $O(k2^{\sqrt{2k}} + n)$ 時間である．

Algorithm 1 $\mathrm{opt}(Y(G)) \leq k$ であるかを判定する

if $m \geq n + k$ **then** “No” を出力して停止．　**end if**
素朴な区間表現 $\mathcal{I} \leftarrow \{I_y : y \in Y(G), d(y) > 1\}$ と摂動された区間表現 $\mathcal{J} \leftarrow \{J_y : y \in Y(G)\}$ および各 $1 \leq t \leq 2|Y(G)|$ において L_t, M_t, R_t を小節 2.2.2 の方法で計算する．
if $|\{\{J_u, J_v\} : J_u, J_v \in \mathcal{J}, J_u \cap J_v \neq \emptyset\}| > k$ **then**
　　“No” を出力して停止．
end if
すべての順序付け可能対 $\{u, v\}$ において $c(u, v), c(v, u)$ を $\mathcal{I}$ を用いて，小節 2.2.3 の方法で計算する．アルゴリズムの途中で NO インスタンスであることがわかった場合は “No” を出力して停止．
$y \leftarrow M_1$ の唯一の元，$c(L_1, \{y\}) \leftarrow 0, \mathrm{opt}(L_1) \leftarrow 0, \mathrm{opt}(L_1 \cup \{y\}) \leftarrow 0$.
for $t = 2$ to $2|Y(G)|$ **do**
　　if ある $y \in Y(G)$ において，$t = a_y$ **then**
　　　　$c(L_t, \{y\}) \leftarrow 0$.
　　　　for $v \in M_t \setminus \{y\}$ **do**
　　　　　　$c(L_t, \{v\}) \leftarrow c(L_{t-1}, \{v\})$.
　　　　end for
　　　　for $S \subseteq M_t$ **do**
　　　　　　if $y \notin S$ **then**
　　　　　　　　$\mathrm{opt}(L_t \cup S) \leftarrow \mathrm{opt}(L_{t-1} \cup S)$.
　　　　　　else
　　　　　　　　$\mathrm{opt}(L_t \cup S) \leftarrow \min_{v \in S}\{\mathrm{opt}(L_t \cup S \setminus \{v\})$
　　　　　　　　　　$+ c(L_t, \{v\}) + c(L_t \cup S \setminus \{v\}, \{v\})\}$.
　　　　　　end if
　　　　end for
　　else
　　　　for $v \in M_t$ **do**
　　　　　　$c(L_t, \{v\}) \leftarrow c(L_{t-1}, \{v\}) + c(y, v)$.
　　　　end for
　　　　for $S \subseteq M_t$ **do**
　　　　　　$\mathrm{opt}(L_t \cup S) \leftarrow \mathrm{opt}(L_{t-1} \cup (S \cup \{y\}))$.
　　　　end for
　　end if
end for
if $\mathrm{opt}(L_{2|Y(G)|}) \leq k$ **then** “Yes” を出力．
else“No” を出力．　**end if**

2.3　計算量の下界

本節では，以下に示す指数時間仮説 [66] が成り立つならば，OSCM に $2^{o(\sqrt{k})}n^{O(1)}$ 時間アルゴリズムが存在しないことを示す．指数時間仮説に基づく計算量の下界はパラメータ化計算量理論や厳密指数時間アルゴリズムの分野で多く知られている．指数時間仮説に関するさらなる情報は [84] のサーベイ論文が詳しい．

指数時間仮説 (ETH)：任意の $t \geq 3$ において，ある定数 c_t が存在し，t-SAT を解く $2^{c_t n}$ 時間アルゴリズムは存在しない．ここで，n は t-CNF 式の変数の数である．

定理 2.3 (Impagliazzo-Paturi-Zane[66])**.** 指数時間仮説が成り立つならば，*3-SAT* をとく $2^{o(m)}$ 時間アルゴリズムが存在しない．ここで m は *3-CNF* 式の節数である．

本結果の目的のために，疎な OSCM の NP 完全性 [89] の証明に使用した 3-SAT から OSCM への多項式時間帰着を記す．多項式時間帰着では以下の問題を用いる．

t-SAT
インスタンス：各節が t 個以下のリテラルを持つような t-CNF 式 ϕ
タスク：ϕ を真にする変数への真理値割り当てが存在するか？

ϕ を真にする変数割り当てが存在するとき，ϕ が**充足可能**であるという．

Vertex Cover
インスタンス：無向グラフ G と整数 k
タスク：G が大きさ k 以下の点カバーを持つか？

$C \subseteq V(G)$ において，G の各辺 $\{u, v\} \in E(G)$ の端点のうち少なくと一方が C に含まれる，つまり $\{u, v\} \cap C \neq \emptyset$ であるとき C を G の**点カバー**という．

Feedback Arc Set
インスタンス：有向グラフ G と整数 k
タスク：G が大きさ k 以下のフィードバック辺集合を持つか？

$F \subseteq E(G)$ において，$(V(G), E(G) \setminus F)$ が閉路を持たないとき F を G の**フィードバック辺集合**という．

補題 2.9 (Karp[69])**.** *n 変数，m 節の 3-CNF 式 ϕ が与えられたとき，ϕ が充足可能であることと G_ϕ に大きさ $n+2m$ の点カバーが存在することが同値になるようなグラフ G_ϕ を出力する多項式時間アルゴリズムが存在する．さらに，$|V(G_\phi)| = 2n+3m$，$|E(G_\phi)| = n+6m$ を満たす．*

証明の概略. $X = \{x_1, x_2, \ldots, x_n\}$ を ϕ の変数集合，$C = \{c_1, c_2, \ldots, c_m\}$ を ϕ の節集合とする．G_ϕ を以下のように構成する．各 $x_i \in X$ において G_ϕ が 2 つの頂点 $u_i, \bar{u}_i$ と辺 $\{u_i, \bar{u}_i\}$ を持つようにする．各 $c_i \in C$ において G_ϕ が長さ 3 の閉路 $\{v_i^1, v_i^2, v_i^3\}$ を持つようにする．また，節 c_j の k 番目のリテラルとして u_i があらわれる場合，辺 $\{u_i, v_j^k\}$ を加え，節 c_j の k 番目のリテラルとして $\bar{u}_i$ があらわれる場合，辺 $\{\bar{u}_i, v_j^k\}$ を加える．明らかに G_ϕ は $2n+3m$ 個の頂点，$n+6m$ 本の辺を持つ．帰着の正しさの証明については [69] を参照．□

補題 2.10 (Karp[69])**.** *n 頂点，m 辺のグラフ G と整数 k が与えられたとき，G が大きさ k の点カバーを持つことと D に大きさ k のフィードバック辺集合が存在することが同値になるような有向グラフ D を出力する多項式時間アルゴリズムが存在する．さらに，$|V(D)| = 2n$，$|E(D)| = n+2m$ を満たす．*

証明の概略. $V(G) = \{u_1, u_2, \ldots, u_n\}$ とする．有向グラフ D を以下のように構成する．各頂点 $u_i \in V(G)$ において D が 2 頂点 v_i, w_i と有向辺 (v_i, w_i) を含むようにする．各辺 $\{u_i, u_j\} \in E(G)$ において，D が有向辺 (w_j, v_i) と (w_i, v_j) を含むようにする．明らかに D は $2n$ 個の頂点，$n+2m$ 本の有向辺を持つ．帰着の正しさの証明については [69] を参照．□

最後に，フィードバック辺集合問題から OSCM への帰着を説明する．Eades-Wormald[45] は OSCM の NP 完全性を示すためにそのような帰着を初めて与えたが，帰着された OSCM インスタンスは $\Theta(nm)$ 辺含んでしまう．ここで，n はフィードバック辺集合問題の頂点数，m は辺数である．定理 2.2 を示すためには，Muñoz-Unfer-Vrt'o[89] の結果を利用する．

補題 2.11 (Muñoz-Unfer-Vrt'o[89])**.** *n 頂点，m 辺の有向グラフ D と整数 k が与えられたとき，D が大きさ k のフィードバック辺集合を持つことと $(G, <_X, k')$ が OSCM の YES インスタンスであることが同値になるよう*

な *OSCM* のインスタンス $(G, <_X, k')$ を出力する多項式時間アルゴリズムが存在する．さらに，$|V(G)| = 5(n+m)$，$|E(G)| = 4(n+m)$ を満たす.

証明の概略. 各 $1 \leq i \leq 5$ において X_i を互いに疎な集合で要素数 $n+m$ であるものとし，ϕ_i を $V(D) \cup E(D)$ から X_i への全単射とする．２部グラフ $G = (X(G), Y(G), E(G))$ を

$$
\begin{aligned}
X(G) &= X_1 \cup X_2 \cup X_3 \cup X_4, \\
Y(G) &= X_5, \\
E(G) &= \{\{\phi_i(w), \phi_5(w) : 1 \leq i \leq 4, w \in V(D) \cup E(D)\}\}
\end{aligned}
$$

とする．明らかに，$|V(G)| = 5(n+m)$，$|E(G)| = 4(n+m)$ である．このとき，OSCM インスタンス $(G, <_X, k')$ を $X(G)$ の全順序 $<_X$ とある整数 k' を用いて定義できる．全順序 $<_X$ と整数 k'，帰着の正しさの証明については [89] を参照. $\square$

これらの帰着をまとめると以下の補題が得られる.

補題 2.12. *n 変数，m 節の 3-CNF 式 ϕ が与えられたとき，ϕ が充足可能であることと $(G, <_X, k)$ が OSCM の YES インスタンスであることが同値になるような OSCM のインスタンス $(G, <_X, k)$ を出力する多項式時間アルゴリズムが存在する．さらに，$|V(G_\phi)| = 40n+105m$，$|E(G_\phi)| = 32n+84m$ を満たす.*

この補題を利用して，定理 2.2 を背理法で示す．OSCM に $2^{o(\sqrt{k})} n^{O(1)}$ 時間アルゴリズムが存在すると仮定する．OSCM のインスタンス $(G, <_X, k)$ において，$k \leq |E(G)|^2$ と仮定できる (そうでなければ，$(G, <_X, k)$ は明らかに YES インスタンスである)．よって，OSCM は $2^{o(|E(G)|)} n^{O(1)}$ 時間で解くことができる．補題 2.12 より，多項式時間で 3-SAT インスタンスから OSCM インスタンスを構成することにより，m を 3-SAT インスタンスの節数とすると，3-SAT を $2^{o(m)} n^{O(1)}$ 時間で解くことができる．これは指数時間仮説に反する.

よって，定理 2.2 が示せた.

2.4　まとめ

本章では，OSCM を解く $O(k2^{\sqrt{2k}} + n)$ 時間アルゴリズムを与えた．この結果の実行時間は，インスタンスサイズに関して最適であり，パラメータに関しては，指数時間仮説が成り立つならば漸近的に最適である.

Fernau ら [47] は OSCM をトーナメント上の整数重み付きフィードバック辺集合問題へ帰着することにより，準指数時間アルゴリズムを与え，Kenyon-Mathieu と Schudy の結果 [72] を用いることで，OSCM に対する多項式時間近似スキームを与えた．興味深い未解決問題として，異なる手法による多項式時間近似スキームの存在が挙げられる．特に，本結果の区間表現を用いてそのようなアルゴリズムの設計が今後の展望である．

OSCM は標準的な動的計画法を用いて $O(2^{|Y(G)|}n)$ 時間で解くことができる．ある $c < 2$ において，OSCM を解く $c^n n^{O(1)}$ 時間アルゴリズムの存在は未解決問題である．

本結果のアルゴリズムは指数領域を必要とする．このアルゴリズムを，"Space versus Time" [81] の議論で多項式領域アルゴリズムにできるかは興味深い問題である．

最後に，本研究結果が実用上も効果的であることを示す必要がある．予備的な実験では，交差数の少ないインスタンスにおいて，非常に高速に動作することがわかった．さらに，この結果を評価するために，[7, 8] などで用いられている階層描画のベンチマークなどで実験する必要がある．

第3章　Two-layer crossing minimization

本章では，Two-Layer Crossing Minimization(TLCM) に対する多項式カーネル化を与える．以下の問題を考える．

> Two-Layer Crossing Minimization(TLCM)
> **インスタンス**：無向２部グラフ $G = (X(G), Y(G), E(G))$ と正整数 k
> **タスク**：$\mathrm{bcr}(\mathcal{D} = (G, <_X, <_Y)) \leq k$ であるような２層描画 $\mathcal{D}$ が存在するか？

頂点が**リーフ**（または**葉**）であるとは，その頂点の次数が 1 であることであり，辺が**リーフ辺**であるとは，その辺がリーフに接続されることである．リーフ辺でない辺を**内部辺**と呼ぶ．辺の重み関数 $w : E(G) \to \mathbb{N}$ が**リーフ重み**であるとは，すべての内部辺の重みが 1 であるときである．

> Leaf-Edge Weighted TLCM(LEW-TLCM)
> **インスタンス**：n 頂点２部グラフ $G = (X(G), Y(G), E(G))$，リーフ重み関数 $w : E(G) \to \mathbb{N}$ と正整数 k
> **タスク**：$\mathrm{bcr}(\mathcal{D} = (G, <_X, <_Y)) \leq k$ であるような２層描画 $\mathcal{D}$ が存在するか？ただし，bcr は重み付き交差数 (交差辺対 $\{e, e'\}$ において，その重みは $w(e)w(e')$ と定義) とする．

LEW-TLCM は明らかに TLCM の一般化である．本章では以下を示す．

定理 3.1. *与えられたグラフが連結であるとき，TLCM に対して $O(k^2)$ 辺の $O(n + k)$ 時間で動作するカーネル化が存在する．*

定理 3.2. *与えられたグラフが連結であるとき，LEW-TLCM に対して $O(k)$ 辺の $O(n + k)$ 時間で動作するカーネル化が存在する．*

上記の定理よりもわずかに弱い定理は，中江の学位論文 [115] に示されているが，本論文では以下の点で異なる．[115] では TLCM に対して $10k^2 + 24k + 7$ 辺のカーネルと，LEW-TLCM に対して $20k + 7$ 辺のカーネルを与えた．本章では TLCM に対して $5k^2 + 14k + 7$ 辺のカーネルと，LEW-TLCM に対して $10k + 7$ 辺のカーネルを与える．さらに，[115] ではカーネル化の実装の詳細を与えていないが，本章では線形時間で動作するカーネル化の実装を与える．いくつかの定義，補題とその証明は [115] と同様であるが，自己完結性のために本章にも記載する．特に，示す補題が [115] と一致している場合やわずかな変更のみを含んでいる場合には，補題に [115] の参照をつける．

3.1 節では TLCM に関する関連研究について述べ，3.2 節では本章で用いる記法や定義を説明する．3.3 節では TLCM および LEW-TLCM に対するカーネル化を与えて，3.4 節でまとめと今後の展望を述べる．

3.1　関連研究

本節では，TLCM に関する先行研究に関して述べる．

TLCM は Garey-Johnson[57] によって NP 完全であることが示された．また，同論文では一般の辺交差数最小化問題の NP 完全性を示す上で，TLCM から一般の辺交差数最小化問題への多項式時間帰着を与えている．[57] の TLCM への帰着は，多重 2 部グラフへの帰着であったが，最近になって，Schaefer[99] は単純 2 部グラフへの帰着を提案し，単純 2 部グラフにおいても TLCM が NP 完全であることが示された．TLCM はグラフが木 [101, 30] や 2 部順列グラフ [103] では多項式時間可解である．また，Shahrokhi ら [101] は，ある条件を満たした 2 部グラフに対して，TLCM に対する $O(\log^2 n)$ 近似アルゴリズムを与えた．

TLCM は以下の h 層交差数最小化問題の特殊なケースである．h-Layer Crossing Minimization は，(2 部グラフとは限らない) n 頂点グラフが与えられ，そのグラフの頂点を h 層に割り当て，各層において頂点順序を並び替えることにより，辺交差数が高々 k であるような階層描画を探す問題である．この問題は，Dujmović ら [38] によって，$h+k$ をパラメータとすると，FPT であることが証明された．[38] のアルゴリズムの実行時間は $2^{O((h+k)^3)}n$ 時間である．また，このアルゴリズムなかでは Bodlaender-Kloks [15, 22] のパス幅決定の固定パラメータアルゴリズムを用いる．これによって得られたパス分解に基づく高度な動的計画法を設計している．

パス幅決定のアルゴリズムとこの動的計画法を実装することは容易ではなく，実用上の観点から，$h = 2$ の限定された問題，つまり TLCM に関してシンプルで高速なアルゴリズムを設計することは有意義である．

TLCM と類似した問題に，Two-Layer Planarization(TLP) がある．この問題は (２部グラフとは限らない)n 頂点グラフが与えられとき，高々k 本の辺を取り除くことにより，グラフを辺交差なしに２層描画できるかを判定する問題である．この問題も NP 完全であることが知られており [44]，[37] は TLP の $O(k)$ 辺カーネル化を与えることで，FPT であることを証明した．また，Suderman [107] は TLP を解く $O(k \cdot 3.562^k + n)$ 時間アルゴリズムを与えた．

TLCM に対する理論的な結果は多いとは言えないが，いくつかの実験的な結果は知られている [25, 29, 68, 93, 110].

本結果は，TLCM に対する初めてのカーネル化を与えた．また，[38] の結果と比較すると，本結果のカーネル化はシンプルで容易に実装が可能である上，その系として得られる $2^{O(k \log k)} + O(n)$ 時間アルゴリズムは，実行時間のパラメータの依存度を大幅に減少させている．本結果のカーネル化は上記の実験的研究への前処理としても利用することができ，[115] では TLCM の整数計画法 [25] を用いて，限定されたインスタンスにおいて本結果のカーネル化が効果的であることを示した．

3.2　諸定義

本章で扱うグラフは無向２部グラフとする．頂点集合の２分割を強調するために，辺集合は $E(G) \subseteq X(G) \times Y(G)$ と見なす場合がある．頂点 v が**リーフ**であるとは，$d(v) = 1$ であるときであり，リーフに接続される辺を**リーフ辺**と呼ぶ．リーフ辺でない辺を**内部辺**と呼ぶ．G の辺 e において，$G - e$ で G から辺 e を取り除いたグラフを表す．**カット頂点** (**橋**) とは，それを取り除くことでグラフの連結成分数が増える頂点 (辺) のことである．**ブロック**とは，カット頂点を含まない極大な連結部分グラフのことである．また，ブロックが**自明なブロック**であるとは，そのブロックが高々２頂点のみからなるときであり，そうでないブロックを**非自明なブロック**と呼ぶ．G' を G の部分グラフであるとすると，$X(G') = X(G) \cap V(G'), Y(G') = Y(G) \cap V(G')$ と表す．

G の**２部交差数** $\mathrm{bcr}(G)$ とは,

$$\mathrm{bcr}(G) = \min_{\pi_X \in \Pi(X(G)), \pi_Y \in \Pi(Y(G))} \mathrm{bcr}(\mathcal{D} = (G, <_{\pi_X}, <_{\pi_Y}))$$

である．$\mathcal{D}$ が $\mathrm{bcr}(\mathcal{D}) = \mathrm{bcr}(G)$ を満たすとき**最適な描画**と呼ぶ.

G の部分グラフ G' と G の２層描画 $\mathcal{D}$ において，$\mathcal{D} \mid G'$ を $\mathcal{D}$ を G' へ制限した２層描画，つまり G' の頂点が $\mathcal{D}$ と同じ順序で並んでいる G' の２層描画とする．異なる頂点 $u, v \in X(G)$ において，$\mathcal{D}$ のなかで u が v の**左** (**右**) にあるとは，$u <_X v (v <_X u)$ を満たすときである．また，$\mathcal{D}$ のなかで $u \in X(G)$ が**最左** (**最右**) であるとは，$u <_X v$ $(v <_X u)$ をすべての $v \in X(G) \setminus \{u\}$ について満たすときである．コンテクストから $\mathcal{D}$ が明らかなときは単に左，右，最左，最右を用い，さらにこれらの用語は $Y(G)$ の頂点に関しても同様に定義する.

3.3　カーネル化

本節では，TLCM に対する $O(k^2)$ 辺のカーネル化を与える．また，同様のアプローチを LEW-TLCM に適用できることを示す.

3.3.1　TLCM に対するカーネル化

本結果のカーネル化は，橋に対する縮約と，リーフ辺の削除からなる．また，２連結成分に関する辺交差数の下界によって，グラフに十分にたくさんの橋が含まれていることを示す.

補題 3.1 (中江 [115])**.** $\mathcal{D}$ を２部グラフ G の２層描画とし，P を $\mathcal{D}$ なかの最右の頂点から最左の頂点へのパスとする．このとき，$V(P)$ に端点を持たない各辺は P の辺と交差する.

証明. $\mathcal{D}$ の組合せ的でない任意の２層描画をひとつ考える．u と v をそれぞれ G の頂点で $\mathcal{D}$ の中で最左と最右の頂点とする．u と v を端点に持ち，G のいずれの頂点とも辺とも接触しない曲線 Q を考える．このとき，Q と P によってできる (自己交差を含むかもしれない) 閉曲線は平面を幾つかの領域に分割する．これらの領域のうちひとつは $X(G) \setminus V(P)$ の頂点をすべて含み，これとはことなる領域で $Y(G) \setminus V(P)$ の頂点すべてを内部に含むものが存在する．よって，Q がいずれの辺とも交差しないこと

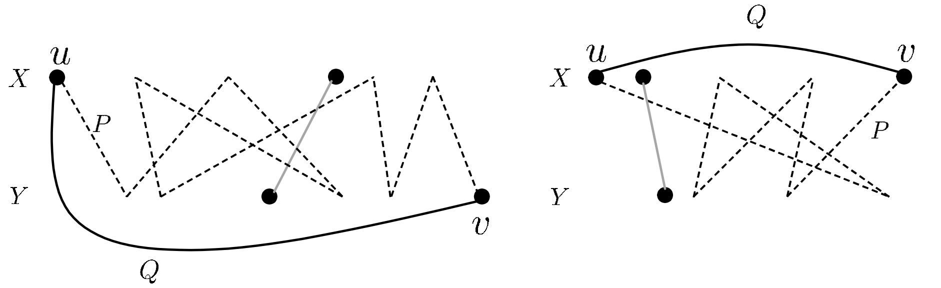

図 3.1: 点折れ線を P，太線を Q とする．左図は P の両端点が異なる層にある場合，右図は同一の層にある場合の例である．$V(P)$ に接しない任意の辺は P の辺を交差しなければならない．

から，$V(P)$ に端点を持たない辺は必ず P の辺と交差しなければならない．図 3.1 はその例である．□

補題 3.2 (中江 [115])**.** G を 2 連結 2 部グラフとすると，$\mathrm{bcr}(G) \geq \frac{|E(G)|-1}{3}$ である．

証明. 辺数の帰納法で証明する．$|E(G)| = 4$ のとき，G は長さ 4 の閉路であり，このとき，任意の 2 層描画はちょうどひとつの交差を持つので，$\mathrm{bcr}(G) \geq \frac{4-1}{3} = 1$.

$|E(G)| > 4$ を仮定する．$\mathcal{D}$ を G の最適な描画とする．以下の 3 つの場合を考える: (1) G が 3 連結，(2) G が閉路，(3) G が 3 連結ではなく，閉路でもない．

(1) e を $\mathcal{D}$ において交差を生じさせている辺とする．G が 3 連結なので，$G-e$ は 2 連結である．帰納法の仮定より，$\mathrm{bcr}(\mathcal{D}) \geq \mathrm{bcr}(\mathcal{D} \mid (G-e))+1 \geq \frac{|E(G-e)|-1}{3} + 1 > \frac{|E(G)|-1}{3}$.

(2) u を $X(G)$ の最左の頂点とする．G は 2 部グラフなので，G は偶閉路である．$|E(G)|/2$ が偶数 (奇数) ならば，v を $Y(G)(X(G))$ の最右の頂点とする．このとき u から v への 2 つのパスは，偶奇性により同じ長さになりえない．P をその 2 つのパスの短い方とすると，$|E(P)| \leq \frac{|E(G)|}{2} - 1$ である．補題 3.1 より， 少なくとも $\frac{|E(G)|}{2} - 1$ 本の辺は $V(P)$ に端点を持たないので，$|E(G)| \geq 5$ の仮定から $\frac{|E(G)|-1}{3}$ 以上の交差を含む.

(3) G が 3 連結ではないので，$G[V(G) \setminus \{u, v\}]$ が非連結となるような頂点 u, v が存在する．$V_1, V_2, \ldots, V_d$ を $G[V(G) \setminus \{u, v\}]$ の連結成分とし，各 i $(1 \leq i \leq d)$ において $G_i = G[V_i \cup \{u, v\}]$ とする．G が閉路ではない

ので，G_i のうち少なくともひとつはパスではなくなるように u, v を選ぶことができる．

はじめに G が辺 $\{u, v\}$ を含む場合を考える．このとき各 i $(1 \le i \le d)$ において G_i は 2 連結であり，帰納法の仮定より，$\mathrm{bcr}(G_i) \ge \frac{1}{3}(|E(G_i)| - 1)$ である．よって，

$$
\begin{aligned}
\mathrm{bcr}(\mathcal{D}) &\ge \sum_{1 \le i \le d} \mathrm{bcr}(\mathcal{D} \mid G_i) \\
&\ge \sum_{1 \le i \le d} \mathrm{bcr}(G_i) \\
&\ge \frac{1}{3} \sum_{1 \le i \le d} (|E(G_i)| - 1) \\
&= \frac{1}{3}(|E(G)| + d - 1 - d) \\
&= \frac{1}{3}(|E(G)| - 1)
\end{aligned}
$$

である．

次に G が辺 $\{u, v\}$ を含まない場合を考える．各 i $(1 \le i \le d)$ において，$\mathcal{B}_i$ を G_i のブロック集合とし，$\mathcal{B} = \bigcup_{1 \le i \le d} \mathcal{B}_i$ とする．各 G_i に辺 $\{u, v\}$ を加えると，2 連結になるので，G_i には $\mathcal{B}_i$ のすべてのブロックを通るような u, v 間のパス P_i が存在する．W_i を $V(P_i) \setminus \{u, v\}$ の部分集合で，G_i のカット頂点をすべて含み，かつ $E(P_i)$ を長さ奇数のパスで分割するような極小頂点集合とする (図 3.2)．P_i のなかで，$W_i \cup \{u, v\}$ の頂点間の各部分パスを 1 本の辺で置き換えたパスを P'_i とし，P'_i の各辺 e において，$\pi(e)$ を P_i の e に置き換えられた部分パスとする．

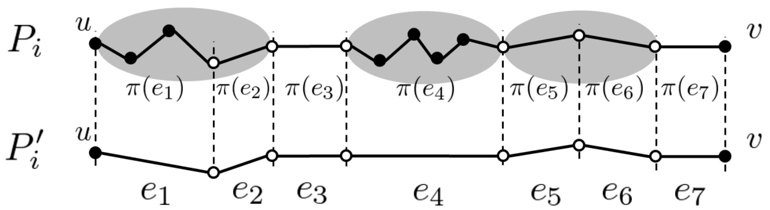

図 3.2: P_i と P'_i の例：灰色の楕円が非自明なブロックを表し，白円は W_i の頂点を表す．$e \in E(P'_i)$ において $\pi(e)$ の長さは奇数である．

H を P'_i の結合して得られるグラフ $H = (\bigcup_{1 \le i \le d} V(P'_i), \bigcup_{1 \le i \le d} E(P'_i))$ とする．ここで，$V(H) \subseteq V(G)$，H は 2 部グラフであり，その頂点集合の

２分割は $(X(G) \cap V(H), Y(G) \cap V(H))$ である．$\mathcal{D}$ のなかの辺交差 (e, e') で辺 e と e' が $\mathcal{B}$ の異なるブロックに属するようなものの集合を Q とする．ここで，$|Q| \geq \mathrm{bcr}(H)$ を示す．$\mathcal{D}'$ を H の２層描画で $\mathcal{D}$ によって誘導されるもの，つまり $\mathcal{D}'$ における $V(H)$ の頂点順序が $\mathcal{D}$ における順序と同じであるような２層描画を考える．$\mathcal{D}'$ において H の辺 e と e' が交差していると仮定する．このとき e と e' の端点の相対的な位置は，パス $\pi(e)$ と $\pi(e')$ の端点の相対的な位置に一致するので，$\mathcal{D}$ において $\pi(e)$ と $\pi(e')$ は交差する．また，それらのパスは点素パスなので，その交差は辺交差である．さらに，$\pi(e)$ と $\pi(e)'$ は $\mathcal{B}$ の異なるブロックに属している (W_i の極小性より，各ブロックに属す頂点でカット頂点ではないものは高々１つしか W_i に選ばれないので，$\pi(e)$ と $\pi(e)'$ が同一のブロックに属しているならば，e と e' は隣接しているので $\mathcal{D}'$ で交差しない)．よって $\mathcal{D}'$ における各交差から Q への単射が存在するので，$|Q| \geq \mathrm{bcr}(\mathcal{D}') \geq \mathrm{bcr}(H)$ である．H が２連結であることと，G_i のうち少なくともひとつはパスではないという仮定より，$|E(H)| < |E(G)|$ を満たし，H へ帰納法を適用すると，$\mathrm{bcr}(H) \geq \frac{|E(H)|-1}{3}$ である．よって $|E(H)| \geq |\mathcal{B}|$ より $\mathrm{bcr}(H) \geq \frac{|\mathcal{B}|-1}{3}$ が導かれる．

補題の不等式を示すことに戻る．$|E(B)| < |E(G)|$ なので，帰納法の仮定より，各 $B \in \mathcal{B}$ において $\mathrm{bcr}(B) \geq \frac{|E(B)|-1}{3}$ である．ここで，この不等式は B が自明なブロックである場合でも成り立つことに注意する．よって，

$$\begin{aligned}
\mathrm{bcr}(G) &= \mathrm{bcr}(\mathcal{D}) \\
&= \sum_{B \in \mathcal{B}} \mathrm{bcr}(\mathcal{D} \mid B) + |Q| \\
&\geq \sum_{B \in \mathcal{B}} \mathrm{bcr}(B) + |Q| \\
&\geq \sum_{B \in \mathcal{B}} \mathrm{bcr}(B) + \mathrm{bcr}(H) \\
&\geq \sum_{B \in \mathcal{B}} \frac{|E(B)|-1}{3} + \frac{|\mathcal{B}|-1}{3} \\
&= \frac{|E(G)|-|\mathcal{B}|}{3} + \frac{|\mathcal{B}|-1}{3} \\
&= \frac{|E(G)|-1}{3}
\end{aligned}$$

であり，補題が成立する．　□

この補題の下界はタイトである．以下はタイトな例である (図 3.3)．この例は任意の $k \geq 1$ において k 交差するような $3k+1$ 辺の 2 連結グラフが存在することを示している．

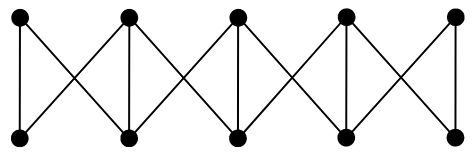

図 3.3: 2 連結グラフ G で $\mathrm{bcr}(G) = \frac{|E(G)|-1}{3}$ を満たす．

以上の補題とその証明は，本質的に中江の学位論文 [115] と同一である．以下からの議論はカーネルの大きさを小さくするために，[115] の解析をさらに精細にしている．そのために，[115] と同じの用語であっても，わずかに異なる使い方をしていることに注意する．

2 層描画 $\mathcal{D}$ において交差 $\{e, e'\}$ が**ブロック** B **に関する局所交差**である (または，単純に**局所交差**) とは，e と e' が B に含まれていることである．そうでない場合は，交差 $\{e, e'\}$ を**非局所交差**と呼ぶ．

補題 3.3. G を 2 部グラフとし，$\mathcal{B}$ を G の非自明なブロック集合とする．このとき，$\mathrm{bcr}(G)$ は少なくとも $\sum_{B \in \mathcal{B}} \frac{|E(B)|-1}{3}$ である．さらに詳細には，G の任意の 2 層描画 $\mathcal{D}$ において，$\mathcal{D}$ は少なくとも $\sum_{B \in \mathcal{B}} \frac{|E(B)|-1}{3}$ の局所交差を含む．

証明. G が 2 つの辺素な部分グラフ G_1 と G_2 を含むとき，$\mathrm{bcr}(G) \geq \mathrm{bcr}(G_1) + \mathrm{bcr}(G_2)$ である．よって補題 3.2 より，$\mathrm{bcr}(G) \geq \sum_{B \in \mathcal{B}} \frac{|E(B)|-1}{3}$ である．また，補題の後半部分は明らかである．　□

TLCM のインスタンス (G, k) を固定し，$k \geq 0$ とする．定理 3.1 を示すために，G は連結であると仮定する．この節の残りにおいて，G が $\mathrm{bcr}(G) \leq k$ ならば，2 部グラフ G' で $\mathrm{bcr}(G) = \mathrm{bcr}(G')$ かつ $|E(G')| \leq f(k)$ を満たすものに変形できることを示す．ここで，$f(k) = O(k^2)$ である (関数 f の詳細は後に示す)．以上を示すことでカーネル化が得られる：もし $|E(G')| > f(k)$ であるならば，(G, k) は NO インスタンスなので，定数サイズの自明な NO インスタンスを出力すればよい．

$LB(G) = \sum_{B \in \mathcal{B}} \frac{|E(B)|-1}{3}$ とする．ここで，$\mathcal{B}$ は G の非自明なブロックの集合であり，$LB(G)$ の値は補題 3.3 の下界である．以下では $k \geq LB(G)$

を仮定する (そうでないならば (G,k) は明らかに NO インスタンスである)．G の橋 e において $G-e$ の 2 つの連結成分がどちらも $k-LB(G)$ よりも多くの辺を含むとき，e は**順序を誘導する**といい，そうでない場合は**順序を誘導しない**という．リーフ辺は常に順序を誘導しないことに注意する．以下の補題は，橋が上記の条件を満たすときに “順序を誘導する” という意味を与える．

補題 3.4 (中江 [115] の変形)**.** $e=(u,v)$ を G の順序を誘導する橋とし，G_1 と G_2 を $G-e$ の連結成分とする．$u \in X(G_1)$，$v \in Y(G_2)$ とする．$\mathrm{bcr}(G) \le k$ ならば，$\mathrm{bcr}(\mathcal{D}) \le k$ であるような 2 層描画 $\mathcal{D}=(G,<_X,<_Y)$ で G_1 の任意の頂点が G_2 の任意の頂点の左にあるようなものが存在する．つまり，任意の $a \in X(G_1), b \in X(G_2)$ において $a <_X b$ であり，任意の $a \in Y(G_1), b \in Y(G_2)$ において $a <_Y b$ である．

証明. $\mathcal{D}'$ を G の 2 層描画で $\mathrm{bcr}(\mathcal{D}') \le k$ であり，G_1 が $Y(G)$ の最左の頂点 y_l を含んでいるものとする．補題3.3 より，$\mathcal{D}'$ は $LB(G)$ 個の局所交差を含む．

まず，$\mathcal{D}'$ における $X(G)$ の最右の頂点 x_r は G_2 に含まれることを示す．そうでないと仮定，つまり G_1 が y_l と x_r の両方を含んでいるとする．このとき補題 3.1 より，G_1 の y_l と x_r 間のパスは $E(G_2)$ の各辺と交差する．また，このパスは $V(G_2)$ と点素なので，それらは非局所交差である．よって $\mathcal{D}'$ は $|E(G_2)|+LB(G)$ を超える交差を含むので $\mathrm{bcr}(\mathcal{D}') \le k$ の仮定に反する．よって，$\mathcal{D}'$ において x_r は G_2 に含まれる．

以下のように 2 層描画 $\mathcal{D}=(G,<_X,<_Y)$ を構成する（図 3.4）:

1. 各 $a \in X(G_1), b \in X(G_2)$ において $a <_X b$,
2. 各 $a \in Y(G_1), b \in Y(G_2)$ において $a <_Y b$,
3. $\mathcal{D}' \mid G_1 = \mathcal{D} \mid G_1$,
4. $\mathcal{D}' \mid G_2 = \mathcal{D} \mid G_2$.

明らかに $<_X$ と $<_Y$ はそれぞれ $X(G)$ と $Y(G)$ の全順序である．以下では $\mathrm{bcr}(\mathcal{D}') \ge \mathrm{bcr}(\mathcal{D})$ を示す．$\mathcal{D}$ において任意の G_1 の辺と任意の G_2 の辺との間に交差はなく，G_1 の部分描画と G_2 の部分描画は保存している．よって上記の不等式を示すためには，$\mathcal{D}$ において e と交差する各 $f \in E(G_1)$ は $\mathcal{D}'$ において $E(G_2) \cup \{e\}$ の辺と少なくともひとつの交差を持つことを示せば良い (対象性により $f \in E(G_2)$ も同様である)．

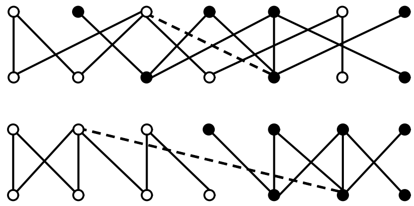

図 3.4: 点線は順序を誘導する橋 e を表し，白点からなるグラフを G_1，黒点からなるグラフを G_2 とする．

$f = (x, y)$ を $\mathcal{D}$ において e と交差する G_1 の辺とする．P を x_r と u 間のパスとする．$e = (u, v)$ と $f = (x, y)$ は $\mathcal{D}$ において交差し，$y \in Y(G_1)$ が $\mathcal{D}$ において $v \in Y(G_2)$ の左にあることから，x は $\mathcal{D}$ において u の右にある. u と x の順序は $\mathcal{D}'$ においても同様である．さらに $x \in X(G_1)$ かつ $x_r \in X(G_2)$ より，$x \neq x_r$ である．よって，補題 3.1 と同様の議論により，f は P のある辺と交差する (図 3.5)．$E(P) \subseteq E(G_2) \cup \{e\}$ より補題が示された．　$\square$

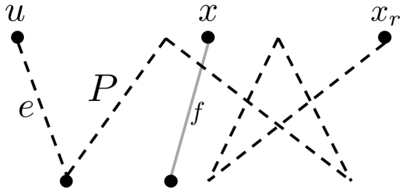

図 3.5: $\mathcal{D}'$ におけるパス P と辺 $f = (x, y)$．P は点線で描かれている．$\mathcal{D}$ と同様に頂点 x は u の右にあり，辺 f は y が v の左右どちらにあっても P の辺と交差する．

順序を誘導する橋 $e = (v_1, v_2)$ が**縮約可能**であるとは以下の条件を満たす：

- 各 i $(i = 1, 2)$ について v_i に e とは異なる順序を誘導する橋 e_i が接続されている，
- 各 v_i $(i = 1, 2)$ に接続される e, e_i を除く辺はすべてリーフ辺である．

G において橋 $e = \{u, v\}$ としたとき，e を**縮約**してできるグラフ H は，$V(H) = V(G) \setminus \{u, v\} \cup \{v_e\}$，$E(H) = E(G[V(G) \setminus \{u, v\}]) \cup \{\{v_e, w\} : w \in N(u) \cup N(v)\}$ とする．また，H は 2 部グラフであることに注意する．さらに，e が縮約可能であるとき，$e'_i = \{v_i, v_e\}$ は H のなかでも順序を誘導する橋である．

補題 3.5 (中江 [115])**.** G が縮約可能な橋 e を持ち，H を G から e を縮約して得られる 2 部グラフとする．このとき $\mathrm{bcr}(G) \leq k$ と $\mathrm{bcr}(H) \leq k$ は同値である．

証明. $e = (v_1, v_2)$ とし，各 i $(i = 1, 2)$ において，e_i を v_i に接続する順序を誘導する橋で，e とは異なるものとする．また，G_i を $G - e_i$ の連結成分で e を含まないものとする．

最初に $\mathrm{bcr}(G) \leq k$ を仮定し，$\mathcal{D}$ を $\mathrm{bcr}(\mathcal{D}) \leq k$ を満たす G の 2 層描画とする．e, e_1, e_2 は順序を誘導する橋なので，補題 3.4 より，$\mathcal{D}$ において G_1 の任意の頂点は v_1 と v_2 の左に位置し，G_2 の任意の頂点は v_1 と v_2 の右に位置する．よって各 i $(i = 1, 2)$ について G_i の辺は $E(G) \setminus (E(G_i) \cup \{e_i\})$ の辺と交差しない．$\mathcal{D}'$ を H の 2 層描画で $\mathcal{D}$ から以下のようににして得られるものとする: $\mathcal{D}$ の部分描画 $\mathcal{D} \mid G[V(G_1) \cup \{v_1\}]$ において，その描画の X 層と Y 層を反転する (X 層にある頂点と Y 層にある頂点を, それらの順序を入れ替えることなく交換する)．このとき v_1 と v_2 は同じ層に位置するが，それらを縮約する．最後に v_1 と v_2 に接続するリーフを G_1 と G_2 の描画を間に配置する (図 3.6)．これらにより，$\mathrm{bcr}(\mathcal{D}') = \mathrm{bcr}(\mathcal{D})$ である．

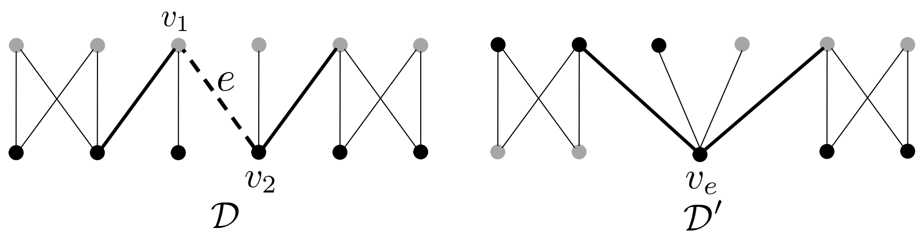

図 3.6: $\mathcal{D}$ と $\mathcal{D}'$ の例 : 点線は縮約可能な橋を表し，太線は順序を誘導する橋である．$\mathcal{D}$ と $\mathcal{D}'$ の交差数は同じである．

補題の逆向きを示すために，$\mathrm{bcr}(H) \leq k$ を仮定し，$\mathcal{D}'$ を H の 2 層描画で $\mathrm{bcr}(\mathcal{D}') \leq k$ を満たすものとする．v_e を H の頂点で e が縮約された

ものとし，各 i $(i=1,2)$ において $e'_i=\{v_i,v_e\}$ とする．e_1 と e_2 が G において順序を誘導する橋であるので, e'_1 と e'_2 も H において順序を誘導する橋である．それにより $\mathcal{D}'$ において $V(G_1)$ に含まれる任意の頂点は v_e と $V(G_2)$ に含まれる任意の頂点の左に位置し, $V(G_2)$ の任意の頂点は v_e と $V(G_1)$ の任意の頂点の右に位置する．よって，上記の $\mathcal{D}$ から $\mathcal{D}'$ への変換の逆向きに行うことで，$\mathrm{bcr}(\mathcal{D})=\mathrm{bcr}(\mathcal{D}')$ なる G の 2 層描画 $\mathcal{D}$ が得られる． □

本結果のカーネル化の最初のステップは，グラフに縮約可能な橋が存在しなくなるまで，その橋を縮約することである．以下の補題でカーネルにおける縮約可能でない非リーフ辺の数の上界を与える．また，以下の補題は中江 [115] の結果をわずかに改善している．

補題 3.6 (中江 [115] の変形)**.** $\mathrm{bcr}(G)\leq k$ と仮定し，G は縮約可能な橋を持たないとする．このとき G の非リーフ辺の本数は高々$5k+3$ である．

この補題の証明は，以下の 2 つの補題から得られる．

補題 3.7 (中江 [115] の変形)**.** $\mathrm{bcr}(G)\leq k$ とすると，順序を誘導しない橋で非リーフ辺であるものの本数は高々$3k-3LB(G)$.

証明. $\mathcal{D}$ を $\mathrm{bcr}(\mathcal{D})\leq k$ を満たす G の 2 層描画とし，パス P を最左の頂点と最右の頂点間のパスする．P に含まれない橋 e でリーフ辺ではないものそれぞれについて，$V(P)$ に点素な辺 f を別々に割り当てることができる : $G-e$ の連結成分で P を含まない部分グラフの辺で，G において e と接する辺を f として割り当てる．補題 3.1 より，それらの辺 f は P の辺と交差するので，高々$k-LB(G)$ 本しか含まれない．P に含まれる橋でリーフ辺ではないものは，P の両端点より $k-LB$ を除いた辺はすべて順序を誘導するので補題が成り立つ． □

補題 3.8 (中江 [115] の変形)**.** $\mathrm{bcr}(G)\leq k$ とすると，順序を誘導する橋で縮約可能でないものの本数は高々$2k+2$.

証明. $\mathcal{D}$ を $\mathrm{bcr}(\mathcal{D})\leq k$ を満たす G の 2 層描画とし，パス P を最左の頂点と最右の頂点間のパスとする．ここで，すべての順序を誘導する橋は P に含まれている : もしそうでないならば，P と点素な $k-LB(G)+1$ 以上の辺が存在し，それらは P の辺と交差し，かつ非局所交差である．

e を順序を誘導する橋で縮約可能でないものとする．このとき e は以下のうち少なくともひとつを満たす : (1) e の端点のうち少なくともひとつ

は順序を誘導しない橋に接続する，(2) e の端点のうち少なくともひとつは非自明はブロックに属する頂点に接続する，または (3) e の各端点は e とは異なる順序を誘導する橋に接続していて，e の端点のうち少なくともひとつは P に含まれない非リーフ辺 f に接続している．

(1) を理由に縮約できない橋は高々 2 本である．G は高々$LB(G)$ 個の非自明なブロックを含むので，(2) を理由に縮約できない橋は高々$2LB(G)$ 本である．(3) については，補題 3.7 の証明と同様の議論を用いて，各 e について $V(P)$ に点素な辺 f を別々に割り当てることができる．f は P の辺と非局所交差を生じさせるので，たかだか $k - LB(G)$ 本しか存在しない．また，各 e は高々 2 本の順序を誘導する橋によってカウントされるので，(3) を理由に縮約できない橋は高々$2k - 2LB(G)$ 本である．

すべてを合計すると高々$2k + 2$ 本の縮約できない順序を誘導する橋が存在する．□

補題 3.3 より，非自明なブロックに属する辺は高々$3LB(G) + 1$ であるので，$LB(G) \leq k$ より，補題 3.6 が得られる．$O(k^2)$ 辺のカーネルを得るために，リーフ辺の数の上界が必要である．

補題 3.9 (中江 [115])**.** 各 $v \in V(G)$ において，$L(v)$ を v に隣接するリーフの集合とし，H を $|L(v)| > k$ を満たすようなすべての $v \in V(G)$ において，$L(v)$ の頂点を $|L(v)| - k - 1$ 個削除して得られるグラフとする．このとき $\mathrm{bcr}(G) \leq k$ と $\mathrm{bcr}(H) \leq k$ は同値.

証明. $\mathcal{D}$ を G の最適な描画とする．このとき，$d(v) > 1$ を満たす各 $v \in V(G)$ において $L(v)$ の頂点は $\mathcal{D}$ のなかで連続して現れると仮定できる．$|L(v)| > k$ とすると，$L(v)$ に含まれるリーフに接続するリーフ辺はすべて交差を持たない (もしそうでないならば，少なくとも $k + 1$ 個の交差を生じさせてしまう)．よって，$L(v)$ の頂点を $|L(v)| - k - 1$ 個削除しても，$L(v) > k$ を満たすので，$\mathrm{bcr}(G) \leq \mathrm{bcr}(H)$ であり，H は G の部分グラフなので，明らかに $\mathrm{bcr}(G) \leq \mathrm{bcr}(H)$ である．□

G が連結なので $|E(G)| + 1 \geq |V(G)|$ と仮定できる．補題 3.6，3.9 より，各頂点は高々$k + 1$ 個のリーフとしか隣接しないので，カーネルの辺数は $5k + 3 + (5k + 4)(k + 1) = 5k^2 + 14k + 7$ である．

G が連結ではないと仮定すると，$\mathrm{bcr}(G) = 0$ であるかは線形時間で判定できる [43] ので，G の各連結成分 C が $\mathrm{bcr}(C) \geq 1$ であることを仮定できる．よって，G の連結成分数は高々k 個なので以下の系が得られる．

系 3.1. *TLCM* に対して $O(k^3)$ 辺のカーネル化が存在する．

3.3.2　LEW-TLCM に対するカーネル化

前小節におけるカーネル化は以下のようにして LEW-TLCM にも適用可能である．G を 2 部グラフ，$w : E(G) \to \mathbb{N}$，整数 k とし，その三組 (G, w, k) を LEW-TLCM のインスタンスとする．$\mathrm{unfold}(G, w)$ を以下の操作によって G から得られる 2 部グラフとする：各 $v \in V(G)$ において，v に接続する重み $w(e)$ のリーフ辺を $w(e)$ 本の重みなのリーフ辺に置き換える．

補題 3.10. (G, w, k) が *LEW-TLCM* の *YES* インスタンスであることと，$(\mathrm{unfold}(G, w), k)$ が *TLCM* の *YES* インスタンスであることは同値.

証明. (G, w) の最適な描画 $\mathcal{D}$ において，G の各リーフ辺 e を $w(e)$ 本のリーフ辺に置き換え，それら $w(e)$ 個の連続したリーフに置き換えられた描画を $\mathcal{D}'$ とする．このとき，$\mathcal{D}$ の重み付き交差数と $\mathcal{D}$ の交差数は等しい.

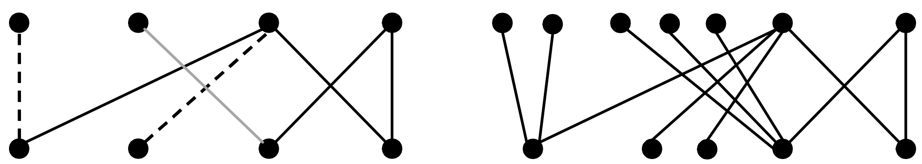

図 3.7:　左図は LEW-TLCM のインスタンス (G, w) とその描画 $\mathcal{D}$．点線で表される辺の重みは 2 とし，灰色線で表される辺の重みを 3 とする．右図は $\mathrm{unfold}(G, w)$ とその描画 $\mathcal{D}'$．$\mathcal{D}$ の重み付き交差数と $\mathcal{D}'$ の交差数は等しい.

また，$\mathrm{unfold}(G, w)$ の最適な描画において同一頂点に接続するリーフは連続して現れると仮定できるので，逆も成り立つ．□

LEW-TLCM のインスタンス (G, w, k) において，$(\mathrm{unfold}(G, w), k)$ に前小節のカーネル化を適用する．得られたカーネル (H, k) に対して，unfold の逆の操作 fold を適用する：つまり，$\mathrm{fold}(H, k)$ は (G', w', k) の三組で，各 $v \in V(H)$ で $d(v) > 1$ を満たすものそれぞれについて，v に接続する l 本のリーフ辺を 1 本の重み $w(e) = l$ を満たすリーフ辺 e に置き換える．この補題と前小節の解析より，このカーネルは高々$5k + 3 + (5k + 4) = 10k + 7$ 本の辺を持つ.

前小節と同様の議論により以下の系を得る.

系 3.2. *LEW-TLCM* に対して $O(k^2)$ 辺のカーネル化が存在する.

3.3.3　アルゴリズムの全体像

定理 3.1 に対するアルゴリズムの全体像を Algorithm 2，Algorithm 3 にまとめる．定理 3.2，系 3.1，系 3.3.3 に対するアルゴリズムも同様に実装可能である．入力される 2 部グラフ G は連結で，n 頂点 m 辺であるとする．G のブロックの集合 $\mathcal{B}$ は Hopcroft-Tarjan のアルゴリズム [65] を用いることにより線形時間で求めることができる．$T_{\mathcal{B}} = (\mathcal{B}, \{\{B, B'\} : B, B' \in \mathcal{B}, B \neq B', V(B) \cap V(B') \neq \emptyset\})$ を頂点集合が $\mathcal{B}$ である木とする．recursive-contraction は $T_{\mathcal{B}}$ を用いて，G の縮約可能な橋を可能な限り縮約して，$\mathrm{bcr}(G) \leq k$ と $\mathrm{bcr}(H) \leq k$ が同値であるようなグラフ H で縮約可能な橋を持たないものを返す.

Algorithm 2 $\mathrm{bcr}(G) \leq k$ であるかを判定する．

Require: G が連結

if $m \geq n + k$ **then**
　自明な NO インスタンスを出力して停止.
end if
$\mathcal{B} \leftarrow G$ のブロックの集合，$LB(G) \leftarrow \sum_{B \in \mathcal{B}} \frac{|E(B)|-1}{3}$.
$T_{\mathcal{B}}$ を作り，r を $T_{\mathcal{B}}$ のあるリーフとする.
$H \leftarrow$ recursive-contraction$(G, T_{\mathcal{B}}, k, r)$
各頂点 $v \in V(H)$ において，$L(v)$ を計算し，$|L(v)| > k$ ならば，$L(v)$ から $|L(v)| - (k+1)$ 頂点を任意に選択し，H から削除する.
if $|E(H)| > 5k^2 + 14k + 7$ **then**
　自明な NO インスタンスを出力.
else
　H を出力.
end if

$B \in \mathcal{B}$ とする．B の重みを $|E(B)|$ と定義する．また，B を根とする $T_{\mathcal{B}}$ の部分木の重みとは，その部分木に含まれる頂点 (ブロック) の重みの総和と定義する．B の子の集合を $c(B)$ と記述する．B が**リーフブロック**であるとは，B が自明なブロックであり，B に含まれるただひとつの辺がリーフ辺のことである．$T_{\mathcal{B}}$ のあるリーフ $r \in V(T_{\mathcal{B}})$ を選ぶと，r は G のリーフブロックもしくは非自明なブロックである．recursive-contraction では r を開始頂点として $T_{\mathcal{B}}$ を後順 (post-order) 深さ優先探索する．このとき，$T_{\mathcal{B}}$ を r を根とした根付き木とみなすことができる．B が**順序を誘**

導する (縮約可能な) ブロックであるとは B が自明なブロックであり，かつ B に含まれる唯一の辺が順序誘導する (縮約可能な) 橋であるときとする．この深さ優先探索で頂点 $B \in V(T_{\mathcal{B}})$ を訪れたとき，以下を計算する：

D1 B を根とする $T_{\mathcal{B}}$ の部分木の重み,

D2 B が順序を誘導ブロックであるどうか,

D3 各 B の子 B' が縮約可能なブロックであるかどうか.

D1 は，$c(B)$ に含まれるブロック B' を根とする $T_{\mathcal{B}}$ の部分木の重みの総和 $+|E(B)|$ なので，動的計画法を用いて全体で $O(n+m)$ 時間で計算可能である．D2 については

- B が自明なブロック,
- (B を根とする部分木の重み)$-1 > k$,
- ($T_{\mathcal{B}}$ の重み)$-$(B を根とする部分木の重み)$> k$,

であるかを判定すればよく，$O(|c(B)|)$ 時間で計算可能であり，全体で $O(n+m)$ 時間で計算可能である．D3 については

- B が順序を誘導するブロック,
- $c(B')$ に順序を誘導するブロック B'' が存在,
- $c(B)$ に含まれるブロックは B' を除いてリーフブロック,
- $c(B')$ に含まれるブロックは B'' を除いてリーフブロック,

であるかを判定すればよく，各ブロック B に $c(B)$ に含まれる順序を誘導するブロック B' へのポインタを持たせることで，$O(|c(B)+c(B')|)$ 時間で計算可能であり，全体で $O(n+m)$ 時間で計算可能である．B' が縮約可能ブロックであるとき $T_{\mathcal{B}}$ から B' を削除し，$c(B')$ を B の子として加える．また D1 をおよび D2 で計算した結果を更新する．

Algorithm 3 では，$T_{\mathcal{B}}$ を後順深さ優先探索をする．B において D1，D2，D3 を計算し，上記の操作を行ったとすると，B を根とする $T_{\mathcal{B}}$ の部分木に含まれる頂点は B を除いてすべて縮約可能なブロックではない．r は縮約可能なブロックではないので，$\mathrm{bcr}(G) \leq k$ ならば，このアルゴリズム

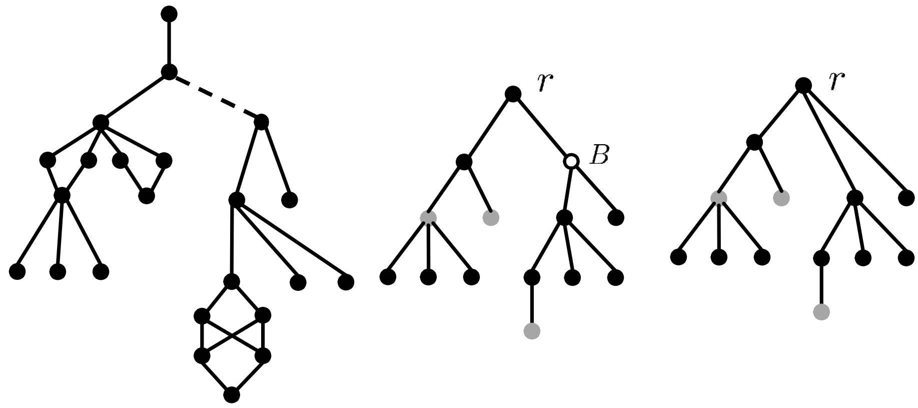

図 3.8: 左図が G の例で中央図がその $T_{\mathcal{B}}$ である．$T_{\mathcal{B}}$ において灰色の頂点が非自明なブロックに対応し，黒色の頂点が自明なブロックと対応する．左図において点線が縮約可能な橋を表し，中央図において白抜きの円がそれに対応する縮約可能なブロック B である．$T_{\mathcal{B}}$ において B へ縮約の操作を適用し，右図の木を得る．

で G から得られる縮約可能な橋を持たないグラフ H は $\mathrm{bcr}(G) = \mathrm{bcr}(H)$ を満たす．

以下の補題を用いることで，TLCM は $2^{O(k \log k)} + O(n)$ 時間で解くことができる．

補題 3.11. *LEW-TLCM* は $2^{O(n \log n)}$ 時間で解くことができる．

証明. 組み合わせ的 2 層描画の数は高々$n!$ であり，ある描画の重み付き辺交差数は多項式時間で計算できるので補題が成り立つ． □

TLCM のインスタンス (G, k) が与えられたとき，LEW-TLCM のインスタンス $\mathrm{fold}(G, k)$ を構成する．系より，連結とは限らない LEW-TLCM に対して高々$k(10k + 7)$ 辺カーネルが存在する．また，そのカーネルの各連結成分は高々$10k + 7$ 本の辺を持つ．$\mathrm{fold}(G, k)$ のカーネルの各連結成分に補題 3.11 のアルゴリズムを適用すると，TLCM は $2^{O(k \log k)} + O(n)$ 時間で解くことができる．

Algorithm 3 recursive-contraction$(G, T_{\mathcal{B}}, k, B)$

$H \leftarrow G$, $w \leftarrow 0$.
for $B' \in c(B)$ **do**
　$H \leftarrow$ recursive-contraction$(H, T_{\mathcal{B}}, k, B')$.
　$w \leftarrow w +$ (B' を根とする $T_{\mathcal{B}}$ の部分木の重み).
end for
if B が順序を誘導するブロック **then**
　if $c(B)$ に含まれる B' が縮約可能なブロック **then**
　　B' を $T_{\mathcal{B}}$ から削除し，$c(B) \leftarrow c(B) \cup c(B') \setminus \{B'\}$ とする．
　　B' に対応する縮約可能な橋を H から縮約する．
　　$w \leftarrow w - 1$.
　end if
end if
$w + |E(B)|$ を B を根とする $T_{\mathcal{B}}$ の部分木の重みとする．
return H.

3.4　まとめ

本章では，TLCM とその一般化である LEW-TLCM に対するカーネル化を与えた．これらのカーネルはグラフが連結ならば，それぞれ高々 $5k^2 + 14k + 7$，$10k^2 + 7$ 本の辺を持つ．グラフが非連結である場合でも，それぞれ $O(k^3)$，$O(k^2)$ 本の辺を持つカーネル化が存在する．これらの結果は TLCM に対する初めての多項式カーネル化である．さらに，TLCM を解く $2^{O(k \log k)} + O(n)$ 時間アルゴリズムを与えた．この系は既存のアルゴリズムの実行時間 $2^{O(k^3)}n$ を大幅に改善している．中江 [115] では，本章と同様の結果を導いたが，本章では

- カーネルサイズの改善
- カーネル化アルゴリズムの実行時間の改善

を果たしている．

以下では TLCM における未解決問題を挙げる．

本結果の改善，つまり TLCM を解く $2^{o(k \log k)} n^{O(1)}$ 時間アルゴリズムの存在は興味深い問題である．特に LEW-TLCM に対する $2^{O(m)} n^{O(1)}$ 時間アルゴリズムが存在すれば，本結果のカーネル化を用いることで，TLCM を解く $2^{O(k)} + O(n)$ 時間アルゴリズムが導かれる．また，TLCM に対する近

似率定数の近似アルゴリズムの存在も未解決である．[58] によると，“Prior to this work, all problems that were known to admit a polynomial kernel, also had approximation algorithms with approximation ratio polynomial in OPT.” と言われており，TLCM に対する $\mathrm{bcr}(G)^{O(1)}$ 近似アルゴリズムの存在が期待される．

第4章　有向グラフのパス幅を計算するアルゴリズム

本章では，有向グラフのパス幅を求める厳密指数時間アルゴリズムを与える．以下の問題を考える．

PATHWIDTH
インスタンス：n 頂点有向グラフ G
タスク：$\mathrm{pw}(G)$ の値

定理 4.1. n 頂点有向グラフのパス幅 $\mathrm{pw}(G)$ は $O(1.89^n)$ 時間で計算可能．

定理 4.1 を実現するアルゴリズムは，補題 1.3 より，無向グラフに対しても適用可能である．

4.1 節では，パス幅を求めることの意義と先行研究について，4.2 節では，本章で用いる定義といくつかの補題を述べる．4.3 節では，4.4 節の理解のために，シンプルな動的計画法の説明をする．4.4 節では，定理 4.1 のアルゴリズムを示し，4.5 節では，本章の結果と今後の展望を述べる．

4.1　研究背景

無向グラフのパス幅や木幅の概念はグラフマイナー理論 [97] やグラフアルゴリズムの分野で重要な役割を果たしている．特にパス幅はグラフ描画問題と関連が深い．

4.1.1　パス幅を用いた最大独立集合問題の解法

与えられたグラフの木幅やパス幅が小さいときに，多くのグラフ上の NP 困難な問題を効率よく解く動的計画法が知られている [4, 11, 13]．最大独立集合問題を例として，パス幅を用いた動的計画法の概略を説明する．

無向グラフ G とし，そのパス分解を $P = (X_1, X_2, \dots, X_t)$ とする．ここでの目的は G の最大独立集合，つまり $I \subseteq V(G)$ で $G[I]$ が辺を持たないような要素数最大の I を見つけることである．

ここで，パス分解 P が**良い** (*nice*) とは $|X_1| = |X_t| = 1$ であり，各 i $(1 < i \leq t)$ において，

(a) $X_{i-1} \subseteq X_i, |X_i \setminus X_{i-1}| = 1$ または,

(b) $X_i \subseteq X_{i-1}, |X_{i-1} \setminus X_i| = 1$

のいずれか一方を満たすときである．(a) の場合 $X_i \setminus X_{i-1} = \{v_i\}$ とし，(b) の場合 $X_{i-1} \setminus X_i = \{v_i\}$ とする．

補題 4.1 (e.g.[76])**.** 無向グラフ G が幅 k のパス分解 P を持つならば，幅 k の良いパス分解 P' を持つ．さらに，G と P が与えられたとき，P' は線形時間で計算可能である．

よって以下では P が良いパス分解であると仮定する．

各 i $(1 \leq i \leq t)$ において，$G_i = G[\bigcup_{j \leq i} X_j]$ とする．$\mathrm{opt}(i, S)$ を G_i において $S \subseteq X_i$ を独立集合に含めたときの最大独立集合の大きさと定義する．つまり

$$\mathrm{opt}(i, S) = \max_{I : E(G[I]) = \emptyset, S \subseteq I} |I| \tag{4.1}$$

である．式 4.1 は以下のように変形できる．$\mathrm{opt}(1, S)$ は S が $\{v_1\}$ または空集合なので，$\mathrm{opt}(1, S) = |S|$ である．各 i $(1 < i \leq t)$，$S \subseteq X_i$ において $\mathrm{opt}(i, S)$ は以下を満たす.

$$\mathrm{opt}(i, S) = \begin{cases} \mathrm{opt}(i-1, S) & (\mathrm{a},\ v_i \notin S) \\ \mathrm{opt}(i-1, S \setminus N(v_i)) + 1 & (\mathrm{a},\ v_i \in S) \\ \max\{\mathrm{opt}(i-1, S), \mathrm{opt}(i-1, S \cup \{v_i\})\} & (\mathrm{b}) \end{cases}$$

だたし，S が G の独立集合でないならば，$\mathrm{opt}(i, S) = -\infty$ である．また最後の式の第 2 項 $\mathrm{opt}(i-1, S \cup \{v_i\})$ は $S \cup \{v_i\}$ が独立集合でないならば，$-\infty$ とする．この再帰式を用いて，動的計画法により $\mathrm{opt}(i, S)$ を計算することで以下が得られる．

補題 4.2. n 頂点無向グラフ G とその幅 k のパス分解が与えられたとき，G の最大独立集合を $O(2^k kn)$ 時間で計算できる．

Bodlaender-Bonsma-Lokshtanov[17] は，最大独立集合問題に対して，補題 4.2 よりも高速な $O(2^k n)$ 時間アルゴリズムを与えた (いろいろな問題に対して，k の多項式因子を取り除いている)．幅が k であるようなパス分解を用いて点カバー問題やハミルトンパス問題など $f(k)n$ 時間で解くことができる．つまり，パス幅をパラメータとするとこれらの問題は FPT である．さらに，パス幅や木幅が定数であるようなグラフにおいて，Courvelle[32] は単項二階述語論理で表現できる問題を線形時間で解くアルゴリズムを与えた．

パス幅を用いたアルゴリズムの設計は，有向グラフにおいては無向グラフほど結果が残されていない．例えば，有向ハミルトンパス問題はパス幅が定数の場合に多項式時間アルゴリズムが存在するが [67]，パス幅をパラメータとしても FPT ではない可能性が高いことが指摘されている [82]．

4.1.2　パス幅とグラフ描画の関係

本小節では，パス幅とグラフ描画の関連性について述べる．Dujmović-Morin-Wood[40] は n 頂点グラフは体積 $O(\mathrm{pw}(G)^2 n)$ の 3 次元直線格子描画ができることを示した．つまり，任意のグラフは 3 次元空間に $O(\mathrm{pw}(G)^2 n)$ 個の格子点があれば，その格子点に頂点を配置することで辺を直線分で描画できることを示した．Hliněný[64] は（一般の）交差数に関して極小なグラフとパス幅の関係性に関して明らかにした．具体的には，交差数が k であるような極小なグラフのパス幅は高々 $2^{6(72\log_2 k+248)k^3+1}-2$ であることを示した．グラフの描画スタイルを多層描画にするとさらなる良い上界が得られる．

Eades-McKay-Wormald[43] は，辺交差がないグラフが 2 層描画を持つのは，グラフの各連結成分が**キャタピラー** (図 4.1) であるときかつそのときに限られることが証明した．

また，連結グラフがキャタピラーであることの必要十分条件はそのパス幅が 1 以下であることが知られている (e.g.[95])．さらに一般的には，辺交差のないグラフが h 層描画を持つならば，そのパス幅は $h-1$ 以下であり，高々 k 個の辺交差を含むならば，そのパス幅は高々 $h+2k-1$ であることが知られている [38](この上界は簡単な議論で $h+k-1$ へ改善可能である)．

パス幅 k の木 [106] や 2 連結外平面グラフ [12] を多層に描画 (ここでの

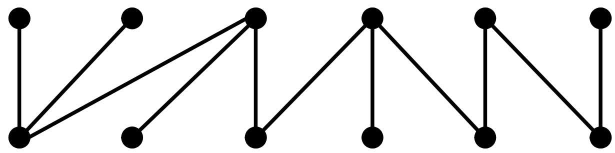

図 4.1: キャタピラーとその2層描画の例．連結グラフがキャタピラーであるとは閉路を含まず，かつあるパスが存在して，すべての頂点がそのパス上に存在するかまたはそのパス上の頂点に隣接するときである．

多層描画は一つの辺がいくつかの層にまたがることを許す) するとき，それぞれ $3k-1$，$4k-3$ 層に交差無しで描画可能である．

4.1.3　先行研究

無向グラフのパス幅を求める問題はNP困難[70]である．さらに，グラフクラスを次数制限のある平面グラフ[86]，弦グラフ[60]，cocomparabilityグラフ[61]，2部distance hereditaryグラフ[78]に制限してもNP困難である．その一方で，順列グラフ[23]，cograph[24]，circular-arcグラフ[104]においては多項式時間可解である．Fomin-Høie[49]は任意の $\epsilon > 0$ において，ある整数 n_ϵ が存在して，$n > n_\epsilon$ である n 頂点3正則グラフのパス幅が $(1/6+\epsilon)n$ 以下であることを示し，そのような幅をもつ多項式時間アルゴリズムを与えた．

近似アルゴリズムの分野では，Bodlaenderら[20]はパス幅を求める問題に対する $O(\log^2 n)$ 近似アルゴリズムを与え，さらに $\mathrm{P} = \mathrm{NP}$ でない限り，ある定数 c が存在して，パス幅を求める $(\mathrm{pw}(G)+c)$ 近似アルゴリズムが存在しないことを示した．Feige-Hajiaghayi-Lee[46]はこの結果を改善し，多項式時間で動作する，幅 $O(\mathrm{pw}(G)\sqrt{\mathrm{pw}(G)}\log n)$ のパス分解を求めるアルゴリズムを与えた．Kloks-Bodlaender[77]はいくつかの理想グラフに対して，パス幅の定数近似アルゴリズムを与えた．

パラメータ化計算量理論の観点からもパス幅を求めるアルゴリズムの研究が行われている．Robertson-Seymourはパス幅が与えられたパラメータ以下であるかを判定する問題がFPTであることを示した[98]．Bodlaender-Kloks[15, 22]はこの問題に対する線形時間FPTアルゴリズムを与えた．また，Bodlaenderら[18]はある計算量理論の予想が成り立つならばこの

問題に対する多項式カーネル化が存在しないことを示した．グラフに大きさ k の点カバーが存在する場合，パス幅は比較的高速に動作する FPT アルゴリズムを持つ．Chappele ら [28] はパス幅を求める $3^k n^{O(1)}$ 時間アルゴリズムを与えた．また，Bodlaender-Jansen-Kratsch[21] は頂点数 $O(k^3)$ のカーネル化を与えた．

これらと比べて，有向グラフのパス幅においては未解決な問題が多い．補題 1.3 より，有向グラフのパス幅を求める問題は NP 困難である．さらに，有向グラフのパス幅が与えられたパラメータ以下であるかを判定する問題が FPT であるかどうかは，著者が知るかぎりではわかっていない．その一方で，この問題に対する XP アルゴリズム，つまりパラメータ k において $n^{O(k)}$ 時間で動作するアルゴリズムは知られている [109, 92]．

厳密指数時間アルゴリズムの分野では，有向および無向グラフのパス幅は頂点順序問題の動的計画法 [19] を用いて $2^n n^{O(1)}$ 時間で求めることができる．この動的計画法の詳細は 4.3 節で述べる．Suchan-Villanger[105] は無向グラフのパス幅を求めるアルゴリズムの実行時間を $O(1.9657^n)$ に改善し，さらに，与えられたグラフ G と十分大きな定数 c に対して幅 $pw(G)+c$ の G のパス分解を求める $O(1.89^n)$ 時間アルゴリズムを与えた．有向グラフのパス幅を求める $c < 2$ なる $c^n n^{O(1)}$ 時間アルゴリズムは，本結果が初めて与えた．

オープンソースソフトウェアの Sage[1] では，有向グラフのパス幅を求める $2^n n^{O(1)}$ 時間の動的計画法と整数計画法の定式化を用いた解法 [102] が実装されている．さらに Biedl ら [3] は整数計画法と SAT を用いて計算機実験を行った．

4.2　準備

いくつかの定義は小節 1.2.2 と重複するが，本節でもう一度定義する．G を有向グラフとする．$\Sigma(G) = \bigcup_{U \subseteq V(G)} \Pi(U)$ とする．$V(\sigma) \cap V(\tau) = \emptyset$ なる 2 つの列 $\sigma, \tau \in \Sigma(G)$ において $\sigma\tau$ を σ の後ろに τ をつなげて得られる $V(\sigma) \cup V(\tau)$ の順列とする．ある τ において $\sigma = \sigma'\tau$ となるとき，σ' を σ の**プレフィックス**と呼び，σ を σ' の**拡張**と呼ぶ．さらに，$|\tau| > 0$ であるとき，σ' を σ の**プロパーな**プレフィックスと呼び，σ を σ' の**プロパーな**拡張と呼ぶ．k を正整数とし，任意の σ のプレフィックス σ' が $|N^-(V(\sigma'))| \leq k$ を満たすとき，σ を G に関して k **分離可能**な列であるという (小節 1.2.2 では $V(G)$ 上の順列に対して，k 分離可能の定義をしたが，本章において

は，$V(G)$ の任意の部分集合上の順列に対して定義している）．σ が G に関して**強 k 分離可能**な列であるとは，σ が $V(\tau) = V(G)$ なる τ のプレフィックスのときである．上記の定義において，グラフがコンテクストから明らかなときは，単純に (強)k 分離可能という．これらの定義は集合上へ拡張できる．各 $U \subseteq V(G)$ において，U が **(強)k 分離可能**であるとは，U 上の順列で (強)k 分離可能であるものが存在することである．G の**頂点分離数** $\mathrm{vs}(G)$ とは，G が k 分離可能な順列を持つような最小の k の値と定義する．

以下の補題は，続く 2 つの補題で用いる．

補題 4.3. $U \subseteq V(G)$ とし，$X \subseteq V(G) \setminus U$ を $N^-(X) \subseteq U \cup N^-(U)$ を満たすものとし，$W = U \cup X$ とする．W が k 分離可能であり U が強 k 分離可能と仮定すると，W も強 k 分離可能である．

証明. U が強 k 分離可能なので，$U = U_0, U_2, \ldots, U_h = V(G)$ なる k 分離可能な集合で，各 i $(1 \leq i \leq h)$ において $U_{i-1} \subseteq U_i$ かつ $|U_i| = |U_{i-1}| + 1$ を満たすものが存在する．各 i $(0 \leq i \leq h)$ において $W_i = U_i \cup X$ とする．補題の仮定より $N^-(X) \subseteq U \cup N^-(U)$ なので，各 i $(0 \leq i \leq h)$ において $N^-(W_i) \subseteq N^-(U_i)$ であり $|N^-(W_i)| \leq k$ である．$W_{i-1} \subseteq W_i$ であり，さらに $|W_i| = |W_{i-1}|$ か $|W_i| = |W_{i-1}| + 1$ のいずれかを満たすので，帰納法の議論により，各 i $(0 \leq i \leq h)$ において W_i は k 分離可能である．よって $W_0 = W$ は強 k 分離可能である．　□

$U \subseteq V(G)$ が (G に関して) **飽和している**とは，$N^-(v) \subseteq U \cup N^-(U)$ を満たす $v \in N^-(U)$ が存在しないことである．各 $U \subseteq V(G)$ において，U の上位集合で飽和しているものが唯一存在し，それを $\mathrm{sat}(U)$ と記述する．つまり，

$$\mathrm{sat}(U) \;=\; U \cup \{v \in N^-(U) \mid N^-(v) \subseteq U \cup N^-(U)\}.$$

ここで $N^-(\mathrm{sat}(U)) \subseteq N^-(U)$ であることに注意する．

補題 4.4. 飽和した集合 $U, W \subseteq V(G)$ において $U \cup N^-(U) = W \cup N^-(W)$ ならば，$U = W$ である．

証明. 飽和した集合 $U, W \subseteq V(G)$ において $X = U \cup N^-(U) = W \cup N^-(W)$ を満たすと仮定する．背理法のため $v \in U \setminus W$ が存在すると仮定する．$v \in U$ より，$N^-(v) \subseteq X$ であり，飽和の定義より，$v \in W$ でなければならないので矛盾する．　□

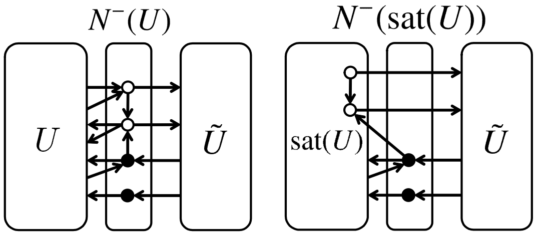

図 4.2: $\mathrm{sat}(U)$ の例．白色頂点 v は $N^-(v) \subseteq U \cup N^-(U)$ を満たすので，$v \in \mathrm{sat}(U)$ である．

無向グラフにおける飽和の概念は [105] において *full set* として利用されている．

補題 4.5. U を $V(G)$ の任意の部分集合とする．U が k 分離可能であるならば $\mathrm{sat}(U)$ も k 分離可能である．さらに，U が強 k 分離可能であるならば $\mathrm{sat}(U)$ も強 k 分離可能である．

証明. $X = \mathrm{sat}(U) \setminus U$ とし，$(v_1, v_2, \ldots, v_{|X|})$ を X の任意の順列とする．$U_0 = U$ とし，各 i $(1 \le i \le |X|)$ において $U_i = U_{i-1} \cup \{v_i\}$ とする．$\mathrm{sat}(U)$ の定義より，各 i $(1 \le i \le h)$ において $N^-(v_i) \subseteq U \cup N^-(U)$ なので，$N^-(U_i) = N^-(U_{i-1}) \setminus \{v_i\}$ である．よって補題の最初主張が成立する．

次に U が強 k 分離可能と仮定する．上述の結果より $\mathrm{sat}(U)$ は k 分離可能である．$N^-(X) \subseteq U \cup N^-(U)$ と補題 4.3 より，$\mathrm{sat}(U) = U \cup X$ は強 k 分離可能である．　□

$U \subseteq V(G)$ とし，$H = G[V(G) \setminus N^-(U)]$ とする．以降では，$\tilde{U} = V(G) \cup (U \cup N^-(U))$ とする．ここで，H の強連結成分 C は $V(C) \subseteq U$ または $V(C) \subseteq \tilde{U}$ のいずれかを満たす．このことは，もしそうでないとき $N^-(U)$ が C の頂点を含まなければならないことからわかる．以下の補題は [105] において *component push rule* として使われたものの有向グラフ版である．

補題 4.6. $U \subseteq V(G)$ とし，$H = G[V(G) \setminus N^-(U)]$ とする．C を H の強連結成分で $V(C) \subseteq \tilde{U}$，$N^-(V(C)) \subseteq U \cup N^-(U)$ かつ $|N^-(U)| + |V(C)| \le k+1$ を満たすものとする．もし U が k 分離可能ならば $U \cup V(C)$ も k 分離可能である．さらに，もし U が強 k 分離可能ならば $U \cup V(C)$ も強 k 分離可能である．

証明. $W = U \cup V(C)$ とする．まず $N^-(W) = N^-(U)$ であることが以下より見て取れる: 補題の仮定より $N^-(V(C)) \subseteq U \cup N^-(U)$ なので $N^-(W) \subseteq N^-(U)$ であり，$C \cap N^-(U) = \emptyset$ より $N^-(U) \subseteq N^-(W)$ である．C の任意の非空部分集合 A において $|N^-(U \cup A)| \leq |N^-(U)| + |V(C)| - 1 \leq k$ なので，補題の最初の主張は成り立つ．

次に U が強 k 分離可能と仮定する．補題の仮定より $N^-(V(C)) \subseteq U \cup N^-(U)$ なので，補題 4.3 より，$W = U \cup V(C)$ は強 k 分離可能である．　□

有向グラフ H において，H の強連結成分の集合を $\mathcal{C}(H)$ と記述する．ここで，$\mathcal{C}(H)$ 上の半順序関係 $\prec$ を $C, D \in \mathcal{C}(H)$ において，C の頂点から D の頂点への有向パスが存在するとき，$C \prec D$ と定義する．$|N^-(U)| \leq k$ なる $U \subseteq V(G)$ において，$H = G[V(G) \setminus N^-(U)]$ とし，$s = k - |N^-(U)| + 1$ としたとき，

$$\begin{aligned}\mathcal{P} = \{C \in \mathcal{C}(H) \quad | \quad & V(C) \subseteq \tilde{U}, |V(C)| \leq s, \text{かつ} \\ & V(D) \subseteq \tilde{U}, |V(D)| > s, D \prec C \text{を満たす} \\ & D \in \mathcal{C}(H) \text{ が存在しない} \}\end{aligned}$$

と定義し，$\mathrm{push}_k(U) = U \cup (\bigcup_{C \in \mathcal{P}} V(C))$ とする．

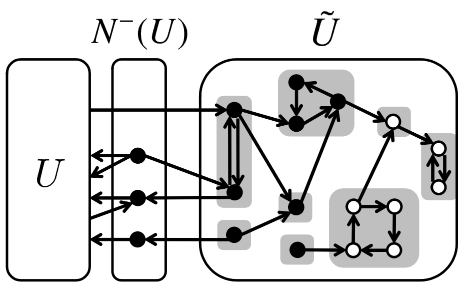

図 4.3: $k = 5, s = 3$ の $\mathrm{push}_4(U)$ の例．$G[\tilde{U}]$ に含まれる強連結成分 (灰色) において，黒色頂点で構成されるものが $\mathcal{P}$ に含まれる．

補題 4.6 を繰り返し適用することで，以下の補題が得られる．

補題 4.7. $U \subseteq V(G)$ とする．U が k 分離可能ならば $\mathrm{push}_k(U)$ も k 分離可能である．さらに，U が強 k 分離可能ならば $\mathrm{push}_k(U)$ も強 k 分離可能である．

無向グラフにおける $\mathrm{push}_k(U)$ の概念は [105] において *component push rule* として利用されている．以下の補題はアルゴリズムの解析において重要である．

補題 4.8. $S \subseteq V(G)$ が $|S| \leq k$ を満たすとする．このとき，$N^-(U) = S$ かつ $\mathrm{push}_k(U) = U$ を満たす $V(G)$ の部分集合 U の数は高々$2^{\frac{n}{s+1}}$ である．ここで，$s = k - |S| + 1$ とする．

証明. $|S| \leq k$ なる $S \subseteq V(G)$ を固定し，$s = k - |S| + 1$ とする．$H = G[V(G) \setminus S]$ とし，$\mathcal{C}_s$ を H の強連結成分で，頂点数が s より大きいものからなる集合とする．ここで，$N^-(U) = S$ を満たす各 $U \subseteq V(G)$ と H の強連結成分 C において，$V(C) \subseteq U$ または $V(C) \subseteq \tilde{U}$ のいずれかであることに注意する．各 $U \subseteq V(G)$ において，$\mathcal{C}_s$ の 2 分割 $(\mathcal{A}_U, \mathcal{B}_U)$ を $V(C) \subseteq U$ ならば $C \in \mathcal{A}_U$，$V(C) \subseteq \tilde{U}$ ならば $C \in \mathcal{B}_U$ と定義する

$\mathcal{C}_s$ の 2 分割 $(\mathcal{A}, \mathcal{B})$ が**妥当**であるとは，$C \in \mathcal{B}$ かつ $D \in \mathcal{A}$ で $C \prec D$ を満たす対 (C, D) が存在しないときである．ここで，$\prec$ は上で用いた半順序とする．ある U から得られる 2 分割 $(\mathcal{A}_U, \mathcal{B}_U)$ に対して，$(\mathcal{A}, \mathcal{B})$ が $\mathcal{A}_U = \mathcal{A}$ かつ $\mathcal{B}_U = \mathcal{B}$ となるためには $(\mathcal{A}, \mathcal{B})$ が $\mathcal{C}_s$ の妥当な 2 分割でなければならない．このことは，$N^-(U) = S$ なる各 $U \subseteq V(G)$ において $C \prec D$，$C \subseteq \tilde{U}$，$D \subseteq U$ の 3 つの条件を満たす H の強連結成分 C, D が存在できないことからわかる．

ある $S \subseteq V(G)$ と $\mathcal{C}_s$ の各妥当な 2 分割 $(\mathcal{A}, \mathcal{B})$ において $N^-(U) = S$, $\mathrm{push}_k(U) = U$，$(\mathcal{A}_U, \mathcal{B}_U) = (\mathcal{A}, \mathcal{B})$ であるような $U \subseteq V(G)$ は高々ひとつ存在する．具体的には，

$$\begin{aligned}\mathcal{D} = \{C \in \mathcal{C}(H) \quad | \quad & |V(C)| \leq s, \text{かつ} \\ & D \in \mathcal{B}, D \prec C \text{を満たす} D \in \mathcal{C}(H) \text{ が存在しない}\}\end{aligned}$$

としたとき，$U = \bigcup_{C \in \mathcal{A} \cup \mathcal{D}} C$ と定義される．

$|\mathcal{C}_s| \leq \frac{n}{s+1}$ なので，$\mathcal{C}_s$ の妥当な 2 分割の数は高々$2^{\frac{n}{s+1}}$ であり，この上限を $N^-(U) = S$ かつ $\mathrm{push}_k(U) = U$ を満たす U の数として適用すると補題が得られる．□

$H(x) = -x \log x - (1-x) \log(1-x), 0 < x < 1$ を 2 進エントロピー関数とする．以下の命題は本章のアルゴリズムの解析のために利用する．

命題 4.1 (e.g. [50])**.** S を n 要素集合とし，$0 < \alpha \leq \frac{1}{2}$ とする．このとき，要素数が高々αn であるような S の部分集合の数は高々$2^{H(\alpha)n}$ である．

4.3　$2^n n^{O(1)}$ 時間アルゴリズム

本節では，有向グラフのパス幅を求める $2^n n^{O(1)}$ 時間アルゴリズムを与える．このアルゴリズムは，続く $O(1.89^n)$ 時間アルゴリズムの基礎を与えている．また，いくつかの 1 グラフレイアウト問題は同様の動的計画法で解くことができる [19].

以下のアルゴリズムは Bellman[9, 10]，Held-Karp[62] による巡回セールスマン問題への動的計画法の応用である．

n 頂点有向グラフ G と整数 k において，$U \subseteq V(G)$ を k 分離可能な非空集合としたとき，ある $v \in U$ で $U \setminus \{v\}$ が k 分離可能であるものが存在する．また，$U \subset V(G)$ が強 k 分離可能であるとき，ある $v \in V(G) \setminus U$ で $U \cup \{v\}$ が強 k 分離可能であるものが存在する．このことから以下のアルゴリズムが導かれる．

Algorithm 4 $\mathrm{vs}(G) \leq k$ であるかを判定する

$\mathcal{U}_0 \leftarrow \{\emptyset\}$，各 i $(1 \leq i \leq n)$ において $\mathcal{U}_i \leftarrow \emptyset$.
for $i = 0$ to $n-1$ **do**
　　for $U \in \mathcal{U}_i$ **do**
　　　　for $v \in V(G) \setminus U : |N^-(U \cup \{v\})| \leq k$ **do**
　　　　　　$\mathcal{U}_{i+1} \leftarrow \mathcal{U}_{i+1} \cup \{U \cup \{v\}\}$.
　　　　end for
　　end for
end for
if $\mathcal{U}_n \neq \emptyset$ **then**
　　YES を出力
else if
　　thenNO を出力
end if

上記の事実より，**Algorithm 4** が正しく動作することがわかる．各 i $(0 \leq i \leq n)$ において $\mathcal{U}_i$ の要素はをアルゴリズムが停止したときの状態であるとすると，**Algorithm 4** の実行時間は $|\bigcup_{0 \leq i \leq n} \mathcal{U}_i| n^{O(1)}$ である．よって，その実行時間は $2^n n^{O(1)}$ である．

次の節では，$|\bigcup_{0 \leq i \leq n} \mathcal{U}_i| = O(1.89^n)$ となるようにアルゴリズムを設計する．

4.4　$O(1.89^n)$ 時間アルゴリズム

本節では n 頂点有向グラフ G と整数 $k \geq 0$ が与えられたとき，$\mathrm{vs}(G) \leq k$ であるかを $O(1.89^n)$ 時間で判定するアルゴリズムを与える．このアルゴリズムを与えることにより，$\mathrm{vs}(G) = O(n)$ であるので定理 4.1 が示される．アルゴリズムは 2 つのアルゴリズム Large-Width と Small-Width からなる．

4.4.1　アルゴリズム Large-width

n 頂点有向グラフ G と整数 $k \geq 0$ を固定する．アルゴリズム Large-Width は $\mathrm{vs}(G) \leq k$ であるかを判定する．

$f : 2^{V(G)} \to 2^{V(G)}$ を $f(U) = \mathrm{sat}(\mathrm{push}_k(U))$ とすると，$U \subseteq f(U)$ より，$f^h(U) = f^{h+1}(U)$ を満たす $h \leq n$ が存在する．このとき，$f^* : 2^{V(G)} \to 2^{V(G)}$ を $f^*(U) = f^h(U)$ と定義する．ここで，ある $U \subseteq V(G)$ において $W = f^*(U)$ ならば $W = \mathrm{sat}(W) = \mathrm{push}_k(W)$ であることに注意する．

Algorithm 5 は $\mathrm{vs}(G) \leq k$ であるかを判定する．

Algorithm 5 Large-width(G, k)

$\mathcal{U}_0 \leftarrow \{f^*(\emptyset)\}$，各 i $(1 \leq i \leq n)$ において $\mathcal{U}_i \leftarrow \emptyset$.
for $i = 0$ to $n-1$ **do**
　for $U \in \mathcal{U}_i$ **do**
　　for $v \in V(G) \setminus U : |N^-(U \cup \{v\})| \leq k$ **do**　▷ (a)
　　　$W \leftarrow f^*(U \cup \{v\})$.
　　　$\mathcal{U}_{|W|} \leftarrow \mathcal{U}_{|W|} \cup \{W\}$.
　　end for
　end for
end for
if $\mathcal{U}_n \neq \emptyset$ **then**
　YES を出力.
else
　NO を出力.
end if

以下の証明では，各 i $(0 \leq i \leq n)$ において $\mathcal{U}_i$ の要素はをアルゴリズムが停止したときの状態であるとする．

補題 4.9. Algorithm 5 は正しく $\mathrm{vs}(G) \le k$ であるかを判定する．

証明. 補題を証明するためには，$V(G)$ が k 分離可能であることと $\mathcal{U}_n$ が非空であることが同値であることを示せば良い．

まず，補題 4.5 と補題 4.7 および (a) の条件より，各 i $(0 \le i \le n)$ において $\mathcal{U}_i$ のすべての要素は k 分離可能である．よって $\mathcal{U}_n \neq \emptyset$ ならば $V(G)$ は k 分離可能である．

補題の逆向きを示すために，$V(G)$ が k 分離可能であると仮定し，$\mathcal{U}_n \neq \emptyset$ であることを示す．t を $\mathcal{U}_i$ が強 k 分離可能な集合を含む最大の i とする．このとき，補題 4.5 と補題 4.7 より，$f^*(\emptyset)$ は強 k 分離可能なので，そのような t を選ぶことができる．$t = n$ ならば補題が成り立つので，$t < n$ と仮定する．$U \in \mathcal{U}_t$ を強 k 分離可能な集合とする．仮定より，ある $v \in V(G) \setminus U$ で $U \cup \{v\}$ が強 k 分離可能であるようなものが存在する．さらに，補題 4.5 と補題 4.7 より，$W = f^*(U \cup \{v\})$ も強 k 分離可能である．アルゴリズムにおいて W は $\mathcal{U}_{|W|}$ に追加され，$|W| > t$ より，t の最大性に反する．よって $\mathcal{U}_n \neq \emptyset$ である．□

以下の補題は k の値がある程度大きい場合に良い実行時間の解析を与える．

補題 4.10. $\delta > 0$ を定数とする．$k > (\frac{1}{3} + \delta)n$ であるとき，Algorithm 5 は $2^{H(\frac{1}{3})n} n^{O(1)}$ 時間で実行される．

証明. $\mathcal{W} = \bigcup_{0 \le i \le n} \mathcal{U}_i$ とする．$\mathcal{W}$ の各要素においてアルゴリズムは $n^{O(1)}$ 時間要するので，補題を示すためには $|\mathcal{W}| = O(2^{H(\frac{1}{3})n})$ であることを示せば良い．

$\mathcal{W}$ の各要素は以下の 3 つの部分集合のうち少なくともひとつに属する:

$$
\begin{aligned}
\mathcal{W}_1 &= \{U \in \mathcal{W} : |U| \le n/3\}, \\
\mathcal{W}_2 &= \{U \in \mathcal{W} : |U \cup N^-(U)| \ge 2n/3\}, \\
\mathcal{W}_3 &= \{U \in \mathcal{W} : |N^-(U)| \le n/3\}.
\end{aligned}
$$

命題 4.1 より，$|\mathcal{W}_1| \le 2^{H(\frac{1}{3})n}$ である．

$\mathcal{W}$ の各要素は飽和しているので，補題 4.4 より，各 $S \subseteq V(G)$ において $U \cup N^-(U) = S$ を満たす $U \in \mathcal{W}$ は高々ひとつである．また $|S| \ge 2n/3$ を満たす異なる S の数は命題 4.1 より高々 $2^{H(\frac{1}{3})n}$ なので $|\mathcal{W}_2| \le 2^{H(\frac{1}{3})n}$ を得る．

$|S| < n/3$ を満たす各 $S \subseteq V(G)$ において，$\mathcal{W}_S = \{U \in \mathcal{W}_3 \mid N^-(U) = S\}$ とする．各 $U \in \mathcal{W}$ において $U = \mathrm{push}_k(U)$ なので補題 4.8 より，各 S に関して $|\mathcal{W}_S| \leq 2^{\frac{n}{k-|S|+2}} \leq 2^{\frac{1}{\delta}}$ である．命題 4.1 より，$|S| < n/3$ を満たす各 $S \subseteq V(G)$ の数は高々 $2^{H(\frac{1}{3})n}$ なので $|\mathcal{W}_3| = 2^{\frac{1}{\delta}} \cdot 2^{H(\frac{1}{3})n} = O(2^{H(\frac{1}{3})n})$ を得る．□

4.4.2　XP アルゴリズム

本小節では Tamaki[109] による有向グラフのパス幅を求める XP アルゴリズムの概略を示す．このアルゴリズムは，続く小節で示すアルゴリズムの重要なサブルーチンを与える．

定理 4.2 (Tamaki[109])**.** n 頂点有向グラフ G と整数 k が与えられたとき，$V(G)$ が k 分離可能であるかを判定する $n^{O(k)}$ 時間アルゴリズムが存在する．

この定理が主張するアルゴリズムは，$\Sigma(G)$ に含まれる k 分離可能な列をノードとする探索木に基づいている．この探索木のノード数は n の階乗に比例する可能性があるので，ノード数を $n^{O(k)}$ するために枝刈りが必要である．以下の補題はその枝刈りを行うために必要である．$\sigma \in \Sigma(G)$ のプロパーな拡張 τ が**非拡大**であるとは，$|N^-(V(\tau))| \leq |N^-(V(\sigma))|$ を満たすときである．

補題 4.11. (Commitment 補題 [109]) $\sigma \in \Sigma(G)$ を強 k 分離可能とし，τ を k 分離可能な σ のプロパーな拡張で非拡大かつ最短のもの，つまり，

1. $|N^-(V(\tau))| \leq |N^-(V(\sigma))|$ かつ

2. 任意の $|\tau'| < |\tau|$ なる σ のプロパーな拡張 τ' において $|N^-(V(\tau'))| > |N^-(V(\sigma))|$

とする．このとき τ も強 k 分離可能である．

補題 4.11 の条件を言い換えれば，強 k 分離可能な列 σ と，その k 分離可能かつ非拡大な拡張 τ において，任意の X $(V(\sigma) \subset X \subset V(\tau))$ が $|N^-(X)| > |N^-(\tau)|$ を満たすことである．

σ を探索木ノードとして現れる列で，非拡大かつプロパーな拡張 τ を持つとする．このとき探索木において，補題 4.11 は σ を根とする探索木の

解が，その子孫 τ を根とする探索木の解によって決定されることを意味する．つまり，σ を根とする探索木において，長さが $|\tau|$ であるような列に対応するノードは τ を除いて探索する必要がない．[109] ではこの補題によって探索木のノード数が $n^{O(k)}$ であることを示した．

この結果を次の小節で示すアルゴリズムで利用するために，探索木の詳細が必要である．σ と τ を $|\sigma| = |\tau|$ を満たす k 分離可能な列とする．σ が τ より**望ましい**とは，$|N^-(V(\sigma))| < |N^-(V(\tau))|$ または $|N^-(V(\sigma))| = |N^-(V(\tau))|$ かつ $\sigma \prec \tau$ を満たすときである．ここで，$\prec$ は $\Sigma(G)$ 上の任意の辞書式順序とする．σ が τ よりも望ましく，σ と τ がある共通のプレフィックス σ' で σ が σ' の最短かつ非拡大である k 分離可能な拡張のとき，σ は τ を**限定する**という．つまり，σ が τ を限定するならば，補題 4.11 より，探索木において τ 以下のノードを探索する必要がなくなる．

各 i $(1 \leq i \leq n)$ において S_i を k 分離可能な列で長さ i のものからなる集合で，

1. $S_1 = \{v \mid |N^-(v)| \leq k\}$.
2. 各 i $(1 \leq i < n)$ において， $T_{i+1} = \{\sigma v \mid \sigma \in S_i, v \in V(G) \setminus V(\sigma), |N^-(V(\sigma v))| \leq k\}$ とし，S_{i+1} は T_{i+1} の部分集合で，T_{i+1} のどの要素によっても限定されない列全体の集合とする．

のように帰納的に定義する．S_i は深さ i の探索木のノード集合とみなす．

S_i の大きさを見積もるために，[109] では各 k 分離可能な列 σ 対して，ある列 $\mathrm{sgn}(\sigma)$ を以下のように定義した．σ の非拡大 k 分離可能な拡張 τ が**局所最短**であるとは，τ のプロパーなプレフィックスで σ の非拡大な拡張が存在しないときである．$\mathrm{sgn}(\sigma)$ を以下のように帰納的に定義する．

1. σ が空ならば $\mathrm{sgn}(\sigma)$ も空.
2. σ が非空かつ，σ のあるプレフィックスの局所最短かつ非拡大な拡張になっているならば，$\mathrm{sgn}(\sigma) = \mathrm{sgn}(\tau)$ である．ここで τ は，σ のプレフィックスで σ が τ の局所最短かつ非拡大な k 分離可能な拡張の中で最短のものとする．
3. それ以外の場合，v を σ の最後の頂点とし，$\sigma = \sigma' v$ としたとき，$\mathrm{sgn}(\sigma) = \mathrm{sgn}(\sigma')v$ とする．

補題 4.12. [109] 各 i $(1 \leq i \leq n)$ において，σ と τ が S_i の異なる要素のとき，$\mathrm{sgn}(\sigma)$ と $\mathrm{sgn}(\tau)$ はどちらも，もう一方のプレフィックスではない．

補題 4.13. $\sigma \in S_{|\sigma|}$ が 1 要素からなる列 v の非拡大な拡張とする．このとき σ は $S_{|\sigma|}$ に含まれる v の唯一の拡張である．

証明. $\sigma^0 = v$ とし，$i = 1, 2, \ldots$ において，σ^i を σ のプレフィックスで σ^{i-1} の非拡大な拡張になっているものの中で最短の列とする．σ が σ^0 の非拡大な拡張なので，σ^1 は一意に定まり，$\sigma^j = \sigma$ となるような $j > 0$ について，各 σ^i $(1 \leq i \leq j)$ が一意に定まる．ここで，各 σ^i $(1 \leq i \leq j)$ は σ^{i-1} の局所最短で非拡大な拡張であり，σ^{i-1} は σ^i の最短のプレフィックス α で，σ^i が α の局所最短かつ非拡大な拡張であることがわかる．よって，sgn の定義より，各 i $(0 \leq i \leq j)$ において $\mathrm{sgn}(\sigma^i) = v$ である．

ここで $S_{|\sigma|}$ が v のある拡張 τ で σ とは異なるものを含んでいると仮定する．補題 4.12 より，$\mathrm{sgn}(\tau)$ の先頭は u $(u \neq v)$ である．これは τ がプレフィックスとして，空列の非拡大な拡張 τ' を含み，そして $N^-(V(\tau')) = \emptyset$ ときのみ起こりうる．もしそのような τ' が存在すると，σ のプレフィックスは限定され，これは σ が $S_{|\sigma|}$ に存在することに反する．よって，σ は $S_{|\sigma|}$ に含まれる v の唯一の拡張である．□

補題 4.14. j を $1 \leq j \leq n$ とし，h を $\bigcup_{1 \leq i \leq j} S_i$ に含まれる列の中で σ で $|N^-(V(\sigma))|$ の最小値とする．このとき，各 $1 \leq i \leq j$ において $|S_i| \leq n^{k-h}$ である．

証明の概略. [109] では，各 k 分離可能な σ において $|\mathrm{sgn}(\sigma)| \leq |N^-(V(\sigma))|$ であることを示した．実際には，各 $\sigma \in \bigcup_{1 \leq i \leq j} S_i$ において，$|\mathrm{sgn}(\sigma)| \leq |N^-(V(\sigma))| - h$ であることを示した．補題 4.12 より，各 i $1 \leq i \leq j$ において $|S_i| \leq n^{k-h}$ が得られる．□

4.4.3　アルゴリズム Small-width

小節 4.4.1 に続き，n 頂点有向グラフ G と整数 k が与えられ，$\mathrm{vs}(G) \leq k$ であるかを判定するアルゴリズムを与える．$\epsilon > 0$ と $d > 1/\epsilon$ を固定する．以下では $k > d$ を仮定する (もしそうでないならば，定理 4.2 より多項式時間で判定可能である)．

アルゴリズム Small-width は $|N^-(U)| < k - d$ かつ $U = \mathrm{push}_k(U)$ を満たす $U \subseteq V(G)$ 上で動的計画法を実行する．$|S| < k - d$ なる各 $S \subseteq V(G)$ において，補題 4.8 より $N^-(U) = S$ かつ $U = \mathrm{push}_k(U)$ を満たす $U \subseteq V(G)$ の数は高々$2^{\epsilon n}$ である．$|N^-(U)| \geq k - d$ を満たす $U \subseteq V(G)$

については，[109] によるアルゴリズムを利用する．以下の補題は，アルゴリズム Small-width のための重要なステップである．

補題 4.15. k 分離可能な集合 $U \subseteq V(G)$ で $k-d \leq |N^-(U)| \leq k$ を満たすものが与えられたとき，$n^{O(d)}$ 時間で

1. U が強 k 分離可能であると判定するか,

2. U が強 k 分離可能でないと判定するか,

3. $U \subset W$ かつ $|N^-(W)| < k-d$ を満たす W で U が強 k 分離可能であることと W が強 k 分離可能であることが同値であるものを出力,

のいずれかを行うアルゴリズムが存在する．

証明. G/U を G から得られるグラフで U をひとつ頂点 v_U に縮約したもの：$V(G/U) = (V(G) \setminus U) \cup \{v_U\}$ かつ,

$$\begin{aligned}
E_1 &= \{(u,v) \in E(G) \mid u,v \in V(G) \setminus U\}, \\
E_2 &= \{(v_U,u) \mid u \in V(G) \setminus U, \exists v \in U : (v,u) \in E(G)\}, \\
E_3 &= \{(u,v_U) \mid u \in V(G) \setminus U, \exists v \in U : (u,v) \in E(G)\}
\end{aligned}$$

としたとき，$E(G/U) = E_1 \cup E_2 \cup E_3$ で定義されるグラフとする．

各 $U \subseteq V(H)$ において $N_H^-(U)$ を H における U の入隣接点集合とする．v_U のみからなる列 v_U を探索木の根に指定して定理 4.2 のアルゴリズムを $H = G/U$ 上で実行する．

もし $|N_H^-(V(\sigma))| < k-d$ を満たす σ が探索木に存在しないならば，探索木のノード数は補題 4.14 より $n^{O(d)}$ であり，よってアルゴリズムの実行時間は $n^{O(d)}$ である．探索木が $V(\sigma) = V(H)$ であるような σ を持っていれば，明らかに v_U が H において強 k 分離可能であり，U が G において強 k 分離可能であることがわかる．そうでないならば，U は G において強 k 分離可能でないと結論付けることができる．

次に，$|N_H^-(V(\sigma))| < k-d$ を満たす σ が探索木に存在したと仮定する．このとき，一般性を失うことなく σ は最短のものと仮定できる．探索木を幅優先順序で探索し，σ が出現したときに停止する．補題 4.13 より，探索木において σ が深さ $|\sigma|$ の唯一のノードである．σ は v_U から，補題 4.11 を適用することで得られるので (補題 4.13 の証明を見よ)，σ が H において強 k 分離可能であることと v_U が H において強 k 分離可能であることが同値である．もとのグラフ G においては， $W = U \cup (V(\sigma) \setminus \{v_U\})$ と

すると U が強 k 分離可能であることと W が強 k 分離可能であることが同値である．このとき，W は 3 番目の場合である．探索木のノード数は $n^{O(d)}$ であり，アルゴリズムの実行時間も $n^{O(d)}$ である．□

以下の証明では，各 i $(0 \leq i \leq n)$ において $\mathcal{U}_i$ の要素はを Algorithm 6 が停止したときの状態であるとする．

補題 4.16. Algorithm 6 は正しく $\mathrm{vs}(G) \leq k$ であるかを判定する．

証明. 補題 4.15 より，(b) で YES と出力することは正しい．よって，アルゴリズムは (c) へ到達したと仮定できる．小節 4.4.1 と同様に，$V(G)$ が k 分離可能であることと $\mathcal{U}_n$ が非空であることが同値であることを示せば良い．

まず $\bigcup_{0 \leq i \leq n} \mathcal{U}_i$ のすべての要素は k 分離可能であることは，補題 4.7，補題 4.15，および (a) の条件よりわかる．よって $\mathcal{U}_n$ が非空ならば $V(G)$ は k 分離可能である．

逆方向を示すために $V(G)$ が k 分離可能であると仮定する．t を $\mathcal{U}_i$ が強 k 分離可能な集合を含むような最大の i とする．補題 4.7 より $\mathrm{push}_k(\emptyset)$ は強 k 分離なので，そのような t は存在する．もし $t = n$ ならば，補題は成り立つので，$t < n$ と仮定する．$U \in \mathcal{U}_t$ を強 k 分離可能な集合とする．U は強 k 分離可能なので，$U' = U \cup \{v\}$ が強 k 分離可能となるような $v \in V(G) \setminus U$ が存在する．$|N^-(U')| < k - d$ ならばアルゴリズムは $W = \mathrm{push}_k(U')$ を $\mathcal{U}_{|W|}$ に加え，補題 4.7 より，W は強 k 分離可能で $|W| > t$ なので矛盾である．また $|N^-(U')| \geq k - d$ ならば，補題 4.15 のアルゴリズムを U' に適用する．U' が強 k 分離可能であるので，補題 4.15 のアルゴリズムは U' が強 k 分離可能であると判定するか，$U \subset W$ かつ $|N^-(W)| < k - d$ である W で強 k 分離可能なものを求める．仮定より，(b) に到達することはないので，そのような W において $\mathrm{push}_k(W)$ が $\mathcal{U}_{|\,\mathrm{push}_k(W)|}$ に加えられる．これは $t < |\,\mathrm{push}_k(W)|$ であるので，t の最大性に反し補題が成り立つ．□

補題 4.17. $k \leq n/2$ において，Algorithm 6 は $O(2^{(H(k/n)+\epsilon)n})$ 時間で実行される．

証明. $\mathcal{U} = \bigcup_{0 \leq i \leq n} \mathcal{U}_i$ とし，$\mathcal{U}$ の各要素を処理するために，アルゴリズムは $n^{O(d)} = n^{O(1)}$ 時間必要なので，補題を示すためには，ある $\epsilon' < \epsilon$ において $|\mathcal{U}| = O(2^{(H(k/n)+\epsilon')n})$ であることを示せば良い．

Algorithm 6 Small-width(G, k)

```
ϵ を選び，$d \leftarrow 1/\epsilon$.
$\mathcal{U}_0 \leftarrow \{\mathrm{push}_k(\emptyset)\}$，各 $i$ $(1 \leq i \leq n)$ において $\mathcal{U}_i \leftarrow \emptyset$.
for $i = 0$ to $n - 1$ do
    for $U \in \mathcal{U}_i$ do
        for $v \in V(G) \setminus U : |N^-(U \cup \{v\})| \leq k$ do        ▷ (a)
            $U' \leftarrow U \cup \{v\}$.
            if If $|N^-(U')| < k - d$ then
                $W \leftarrow \mathrm{push}_k(U')$.
                $\mathcal{U}_{|W|} := \mathcal{U}_{|W|} \cup \{W\}$.
            else
                補題 4.15 のアルゴリズムを $U'$ に適用.
                if $U'$ が強 $k$ 分離可能 then
                    YES を出力して停止.        ▷ (b)
                else if $U'$ が強 $k$ 分離可能ではない then
                    なにもしない.
                else
                    $W \leftarrow$ 補題 4.15 のアルゴリズムの返り値.
                    $\mathcal{U}_{|\mathrm{push}_k(W)|} := \mathcal{U}_{|\mathrm{push}_k(W)|} \cup \{\mathrm{push}_k(W)\}$.
                end if
            end if
        end for
    end for
end for
if $\mathcal{U}_n \neq \emptyset$ then        ▷ (c)
    YES を出力.
else
    NO を出力.
end if
```

$U \in \mathcal{U}$ とする．補題 4.8 より，$|S| < k - d$ なる各 $S \subseteq V(G)$ において，$N^-(U) = S$ かつ $\mathrm{push}_k(U) = U$ を満たす U の数は高々$2^{\epsilon' n}$ である．ここで，$\epsilon' = \frac{1}{d+2} < \epsilon$ とする．$\mathcal{U}$ の各要素 U は $\mathrm{push}_k(U) = U$ かつ $|N^-(U)| < k-d$ を満たすので，$|\mathcal{U}| \leq 2^{H(k/n)n} \cdot 2^{\epsilon' n}$ である． □

4.4.4　アルゴリズムの全体像

有向グラフ G と整数 k が与えられたとき，定理 4.1 は LARGE-WIDTH(G, k) と SMALL-WIDTH(G, k) を組み合わせることで得られる．ここで $2^{H(1/3)} < 1.89$ であり，δ と ϵ を $2^{H(1/3+\delta)+\epsilon} < 1.89$ を満たすように選ぶ．$k > (\frac{1}{3} + \delta)n$ ならば LARGE-WIDTH(G, k) を適用し，そうでないならば，SMALL-WIDTH(G, k) をを適用する．補題 4.10 と補題 4.17 より，実行時間は $O(1.89^n)$ である．

4.5　まとめ

本章では，n 頂点有向グラフのパス幅を求める $O(1.89^n)$ 時間アルゴリズムを与えた．この結果は Suchan-Villanger[105] の結果を高速化かつ有向グラフへ一般化と 2 方向に改善している．

本結果のアイデアは，4.3 節で述べた頂点順序を求める動的計画法をある “特殊な” 頂点集合上で行うことで計算量を改善した．このようなアイデアを**カット幅問題**に適用できるかは興味深い問題である (カット幅問題への $\epsilon > 0$ において $(2-\epsilon)^n n^{O(1)}$ 時間アルゴリズムが存在するかは，この分野での大きな未解決問題として知られている)．また，無向グラフの木幅を求める現在最も高速な厳密指数時間アルゴリズムのアイデア は本質的に本結果とは異なる．この結果は Cops and Robber game による特徴付けを [100] 用いて，実行時間が minimal separator よ potential maximal clique の数に比例し，その実行時間は $O(1.7347^n)$ である [52]．パス幅もゲームに基づく特徴付け [74] がなされているにもかかわらず，同様のアイデアに基づくアルゴリズムは知られていない．

有向グラフのパス幅が，与えられたパラメータ以下であるかを判定する問題が FPT になるかどうかは重要な未解決問題である．その一方で有向グラフをトーナメント (とその一般化) に限ると FPT であることが知られている [51].

謝辞

まず初めに，指導教員の明治大学理工学部情報科学科玉木久夫教授に深く感謝します．玉木教授には研究のための知識や方法など，あらゆる面で大変お世話になりました．本学位請求論文が完成したのも，玉木教授のおかげと言っても過言ではありません．学部３年後期からの６年半の間，研究室での生活や進路などにおいても，サポートしていただきました．学位請求論文作成にあたっては，同学科の石畑清教授，疋田輝雄教授にも貴重なご意見を頂きました．お礼申し上げます．また，論文共著者の皆様にも感謝致します．大阪大学大学院情報科学研究科の梅谷俊治准教授には，公私に渡って多くのアドバイスを頂きました．研究会などに誘っていただいたことで，私の視野を広げる大きなチャンスを下さったこと，感謝致します．計算理論研究室のメンバーおよびOBの方々には，研究室の生活の上でのサポートや研究に関するアイデアを多く頂きました．最後に，私の両親と兄弟には私生活の面で，たくさんの迷惑をかけてしまいました．辛抱強く見守っていただいたこと，深く感謝致します．

参考文献

[1] Sage: Open source mathematics software. `http://www.sagemath.org`. Accessed: 2013-12-19.

[2] N. Alon, D. Lokshtanov, and S. Saurabh. Fast fast. In S. Albers, A. Marchetti-Spaccamela, Y. Matias, S. Nikoletseas, and W. Thomas, editors, *Proceedings of the 36th International Colloquium on Automata, Languages and Programming*, volume 5555 of *Lecture Notes in Computer Science, ICALP'09*, pages 49–58. Springer, 2009.

[3] T. Biedl amd T. Bläsius, B. Niedermann, M. Nöllenburg, R. Prutkin, and I. Rutter. Using ilp/sat to determine pathwidth, visibility representations, and other grid-based graph drawings. In S. Wismath and A. Wolff, editors, *Proceedings of the 21st International Symposium on Graph Drawing*, volume 8242 of *Lecture Notes in Computer Science, GD'94*, pages 460–471. Springer, 2013.

[4] S. Arnborg and A. Proskurowski. Linear time algorithms for np-hard problems restricted to partial k-trees. *Discrete Applied Mathematics*, 23(1):11–24, 1989.

[5] J. Barát. Directed path-width and monotonicity in digraph searching. *Graphs and Combinatorics*, 22(2):161–172, 2006.

[6] G. D. Battista, P. Eades, R. Tamassia, and I. G. Tollis. *Graph drawing: Algorithms for the visualization of graphs.* Prentice Hall, 1998.

[7] G. D. Battista, A. Garg, G. Liotta, A. Parise, R. Tamassia, E. Tassinari, F. Vargiu, and L. Vismara. Drawing directed acyclic graphs:

An experimental study. *International Journal of Computational Geometry and Applications*, 10(6):623–648, 2000.

[8] G. D. Battista, A. Garg, G. Liotta, R. Tamassia, E. Tassinari, and F. Vargiu. An experimantal comparison of four graph drawing algorithm. *Computational Geometry: Theory and Applications*, 7(5-6):303–325, 1997.

[9] R. Bellman. Combinatorial processes and dynamic programming. In *Proceedings of the 10th Symposium in Applied Mathematics*, volume 10, pages 217–249. Ametican Mathematical Society, 1960.

[10] R. Bellman. Dymamic programming treatment of the traveling salesman problem. *Journal of the ACM*, 9:61–63, 1962.

[11] M. W. Bern, E. L. Lawler, and A. L. Wong. Linear-time computation of optimal subgraphs of decomposable graphs. *Journal of Algorithms*, 8(2):216–236, 1987.

[12] T. Biedl. A 4-approximation for the height of drawing 2-connected outer-planar graphs. In T. Erlebach and G. Persiano, editors, *Proceedings of the 10th Internatinal Workshop on Approximation and Online Algorithms*, volume 7846 of *Lecture Notes in Computer Science, WAOA'12*, pages 272–285. Springer, 2013.

[13] H. L. Bodlaender. Dynamic programming on graphs with bounded treewidth. In T. Lepistö and A. Salomaa, editors, *Proceedings of the 15th International Colloquium on Automata, Languages and Programming*, volume 317 of *Lecture Notes in Computer Science, ICALP'88*, pages 105–118. Springer, 1988.

[14] H. L. Bodlaender. A tourist guide through treewidth. *Acta Cybernetica*, 11:1–23, 1993.

[15] H. L. Bodlaender. A linear-time algorithm for finding tree-decompositions of small treewidth. *SIAM Journal on Computing*, 25(6):1305–1317, 1996.

[16] H. L. Bodlaender. A partial k-arboretum of graphs with bounded treewidth. *Theoretical Computer Science*, 209(1–2):1–45, 1998.

[17] H. L. Bodlaender, P. Bonsma, and D. Lokshtanov. The fine details of fast dynamic programming over tree decomposistions. In G. Gutin and S. Szeider, editors, *Proceedings of the 8th International Symposium on Parameterized and Exact Computation*, volume 8246 of *Lecture Notes in Computer Science, IPEC'13*, pages 41–53. Springer, 2013.

[18] H. L. Bodlaender, R. G. Downey, M. R. Fellows, and D. Hermelin. On problems without polynomial kernels. *Journal of Computer and System Sciences*, 75(8):423–434, 2009.

[19] H. L. Bodlaender, F. V. Fomin, A. M. C. A. Koster, D. Kratsch, and D. M. Thilikos. A note on exact algorithms for vertex ordering problems of graphs. *Theory of Computing Systems*, 50(3):420–432, 2012.

[20] H. L. Bodlaender, J. R. Gilbert, H. Hafsteinsson, and T. Kloks. Approximating treewidth, pathwidth, and minimum elimination tree height. In G. Schmidt and R. Berghammer, editors, *Proceedings of the 17th International Workshop on Graph-Theoretic Concepts in Computer Science*, volume 570 of *Lecture Notes in Computer Science, WG'91*, pages 1–12. Springer, 1992.

[21] H. L. Bodlaender, B. M. P. Jansen, and S. Kratsch. Kernel bounds for structural parameterizations of pathwidth. In F. V. Fomin and P. Kaski, editors, *Proceedings of the 13th Scandinavian Symposium and Workshops on Algorithm Theory*, volume 7357 of *Lecture Notes in Computer Science, SWAT'12*, pages 352–363. Springer, 2012.

[22] H. L. Bodlaender and T. Kloks. Efficient and constructive algorithms for the pathwidth and treewidth of graphs. *Journal of Algorithms*, 21(2):358–402, 1996.

[23] H. L. Bodlaender, T. Kloks, and D. Kratsch. Treewidth and pathwidth of permutation graphs. In A. Lingas, S. Carlsson, and R. Karlsson, editors, *Proceedings of the 20th International Colloquium on Automata, Languages and Programming*, volume 700 of *Lecture Notes in Computer Science, ICALP'93*, pages 114–125. Springer, 1993.

[24] H. L. Bodlaender and R. H. Möhring. The pathwidth and treewidth of cographs. *SIAM Journal on Discrete Mathematics*, 6(2):181–188, 1992.

[25] C. Buchheim, A. Wiegele, and L. Zheng. Exact algorithms for the quadratic linear ordering problem. *INFORMS Journal on Computing*, 22(1):168–177, 2009.

[26] L. Cai, J. Chen, R. G. Downey, and M. R. Fellows. Advice classes of parameterized travtability. *Annals of Pure and Applied Logic*, 84:119–138, 1997.

[27] T. Catarci. The assignment heuristics for crossing reduction. *IEEE Transactions on Systems, Man, and Cybernetics*, 25(3):515–521, 1995.

[28] M. Chappele, M. Liedloff, I. Todinca, and Y. Villanger. Treewidth and pathwidth parameterized by the vertex cover number. In F. Dehne, R. Solos-Oba, and J.-R. Sack, editors, *Proceedings of the 13th Algorithms and Data Structures Symposium*, volume 8037 of *Lecture Notes in Computer Science, WADS'13*, pages 232–243. Springer, 2013.

[29] M. Chimani, P. Hungerländer, M. Jünger, and P. Mutzel. An sdp approach to multi-level crossing minimization. *Journal of Experimental Algorithmics*, 17:1–16, 2012.

[30] F. R. K. Chung. On optimal linear arrangements of trees. *Computers and Mathematics with Applications*, 10(1):43–60, 1984.

[31] T. H. Cormen, C. E. Leiserson, R. L. Rivest, and C. Stein. *Introduction to algorithms*. The MIT Press, 3rd edition, 2009.

[32] B. Courcelle. The monadic second-order logic of graphs iii: Tree-decompositions, minor and complexity issues. *Informatique Théorique et Applications*, 26:257–286, 1992.

[33] J. Díaz, J. Petit, and M. Serna. A survey of graph layout problems. *ACM Computing Surveys*, 34(3):313–356, 2002.

[34] R. Diestel. *Graph Theory*. Springer, 4th edition, 2010.

[35] R. G. Downey and M. R. Fellows. *Parameterized Complexity*. Springer, 1998.

[36] S. Dresbach. A new heuristic layout algorithm for directed acyclic graphs. In A. Bachem and A. Drexl, editors, *Operations Research Proceedings 1994*, pages 121–126, 1994.

[37] V. Dujmović, M. R. Fellows, M. Hallet, M. Kitching, G. Liotta, C. McCartin, N. Nishimura, P. Ragde, F. Rosamond, S. Whitesides, and D. R. Wood. A fixed parameter approach to two-layer planarization. *Algorithmica*, 45(2):159–182, 2006.

[38] V. Dujmović, M. R. Fellows, M. Kitching, G. Liotta, C. McCartin, N. Nishimura, P. Ragde, F. Rosamond, M. Suderman, S. Whitesides, and D. R. Wood. On the parameterized complexity of layered graph drawing. *Algorithmica*, 52(2):267–292, 2008.

[39] V. Dujmović, H. Fernau, and M. Kaufmann. Fixed parameter algorithms for one-sided crossing minimization revisited. *Journal on Discrete Algorithms*, 6(2):313–323, 2008.

[40] V. Dujmović, P. Morin, and D. R. Wood. Path-width and three-dimensional straight-line grid drawings of graphs. In M. T. Goodrich and S. G. Kobourov, editors, *Proceedings of the 10th International Symposium on Graph Drawing*, volume 2528 of *Lecture Notes in Computer Science, GD'02*, pages 42–53. Springer, 2002.

[41] V. Dujmović and S. Whitesides. Efficient fixed parameter tractable algorithm for 1-sided crossing minimization. *Algorithmica*, 40(1):15–31, 2004.

[42] P. Eades and D. Kelly. Heuristics for reducing crossings in 2-layered networks. *Ars Combinatoria*, 21(A):89–98, 1986.

[43] P. Eades, B. D. McKay, and N. C. Wormald. On an edge crossing problem. In *Proceedings of the 9th Australian Computer Science Conference*, pages 327–403, 1986.

[44] P. Eades and S. Whitesides. Drawing graphs in two layers. *Theoretical Computer Science*, 131(2):361–374, 1994.

[45] P. Eades and N. C. Wormald. Edge crossings in drawings of bipartite graphs. *Algorithmica*, 11(4):379–403, 1994.

[46] U. Feige, M.T. Hajiaghayi, and J. R. Lee. Improved approximation algorithms for minimum-weight vertex separators. In *Proceedings of the 37th Annual ACM Symposium on Theory of Computing*, STOC'05, pages 563–572, 2005.

[47] H. Fernau, F. V. Fomin, D. Lokshtanov, M. Mnich, G. Philip, and S. Saurabh. Ranking and drawing in subexponential time. In C. S. Iliopoulos and W. F. Smyth, editors, *Proceedings of the 21st International Workshop on Combinatorial Algorithms*, volume 6460 of *Lecture Notes in Computer Science, IWOCA'10*, pages 337–348. Springer, 2010.

[48] J. Flum and M. Grohe. *Parameterized complexity theory.* Springer, 2006.

[49] F. V. Fomin and K. Høie. Pathwidth of cubic graphs and exact algorithms. *Information Processing Letters*, 97(5):191–196, 2006.

[50] F. V. Fomin and D. Kratsch. *Exact exponential algorithms.* Springer, 2010.

[51] F. V. Fomin and M. Pilipczuk. Jungles, bundles, and fixed parameter tractability. In *Proceedings of the 24th Annual ACM-SIAM Symposium on Discrete Algorithms*, SODA'13, pages 396–413. SIAM, 2013.

[52] F. V. Fomin and Y. Villanger. Finding induced subgraphs via minimal triangulations. In *Proceedings of the 27th International Symposium on Theoretical Aspects of Computer Science*, STACS'10, pages 383–394, 2010.

[53] M. Fröhlich and M. Werner. Demonstration of the interactive graph visualization system davinci. In R. Tamassia and I. Tollis, editors, *Proceedings of DIMACS Workshop on Graph Drawing*, volume

894 of *Lecture Notes in Computer Science, GD'94*, pages 274–277. Springer, 1995.

[54] E. R. Gansner, E. Koutsofios, and S. C. North. Drawing graph with *dot*. Technical report, AT&T Labs, 2002.

[55] E. R. Gansner and S. C. North. An open graph visualization system and its applications to software engineering. *Software Practice & Experience*, 30(11):1203–1233, Sep. 2000.

[56] M. R. Garey and D. S. Johnson. *Computers and Intractability: A Guide to the Theory of NP-Completeness*. W. H. Freeman and Compan, 1979.

[57] M. R. Garey and D. S. Johnson. Crossing number is np-complete. *SIAM Journal on Algebraic and Discrete Methods*, 4(3):312–316, 1982.

[58] A. C. Giannopoulou, D. Lokshtanov, S. Saurabh, and O. Suchý. Tree deletion set has a polynomial kernel (but no opt$^{O(1)}$ approximation). *CoRR*, abs/1309.7891, 2013.

[59] J. Guo and R. Niedermeier. Invitation to data reduction and problem kernelization. *ACM SIGACT News*, 38:31–45, 2007.

[60] J. Gusted. On the pathwidth of chordal graphs. *Discrete Applied Mathematics*, 45(3):233–248, 1993.

[61] M. Habib and R. H. Möhring. Treewidth of cocomparability graphs and a new order-theoretic parameter. *Order*, 11(1):47–60, 1994.

[62] M. Held and R. M. Karp. A dynamic programming approach to sequencing problems. *SIAM Journal on Applied Mathematics*, 10:196–210, 1962.

[63] M. Himsolt. Graphlet: design and implementation of a graph editor. *Software Practice & Experience*, 30(11):1303–1324, Sep. 2000.

[64] P. Hliněný. Crossing-number critical graphs have bounded pathwidth. *Journal of Combinatorial Theory, Series B*, 88(2):347–367, 2003.

[65] J. E. Hopcroft and R. E. Tarjan. Efficient algorithms for graph manipulation. *Communications of the ACM*, 16(6):372–378, Jun. 1973.

[66] R. Impagliazzo, R. Paturi, and F. Zane. Which problems have strongly exponential complexity? *Journal of Computer and System Sciences*, 63(4):512–530, 2001.

[67] T. Johnson, N. Robertson, P. D. Seymour, and R. Thomas. Directed tree-width. *Journal of Combinatorial Theory, Series B*, 82(1):138–154, 2001.

[68] M. Jünger and P. Mutzel. 2-layer straightline crossing minimization: Performance of exact and heuristic algorithms. *Journal of Graph Algorithms and Applications*, 1(1):1–25, 1997.

[69] R. M. Karp. Reducibility among combinatorial problems. In *Complexity of Computer Computations*, pages 85–103. Plenum Press, New York, 1972.

[70] T. Kashiwabara and T. Fujisawa. Np-completeness of the problem of finding a minimum-clique-number interval graph containing a given graph as a subgraph. In *Proceedings of International Symposium on Curcuits and Systems*, pages 657–660, 1979.

[71] M. Kaufmann and D. Wagner, editors. *Drawing graphs: Methods and models*, volume 2025 of *Lectute Notes in Computer Science*. Springer, 2001.

[72] C. Kenyon-Mathieu and W. Schudy. How to rank with few errors. In *Proceedings of the 39th Annual ACM Symposium on Theory of Computing*, STOC'07, pages 95–103. ACM, 2007.

[73] G. N. Kinnersley. The vertex separation number of a graph equals its path-width. *Information Processing Letters*, 42(6):345–350, 1992.

[74] L. M. Kirousis and C. H. Papadimitriou. Interval graphs and searching. *Discrete Mathematics*, 55(2):181–184, 1985.

[75] K. Kitsunai, Y. Kobayashi, K. Komuro, T. Tano, and H. Tamaki. Computing directed pathwidth in $O(1.89^n)$ time. In D. M. Thilikos and G. J. Woeginger, editors, *Proceedings of the 7th International Symosium on Parameterized and Exact Computation*, volume 7535 of *Lecture Notes in Computer Science, IPEC'12*, pages 182–193. Springer, 2012.

[76] T. Kloks, editor. *Treewidth–Computations and Approximations*, volume 842 of *Lectute Notes in Computer Science*. Springer, 1994.

[77] T. Kloks and H. L. Bodlaender. Approximating treewidth and pathwidth of some classes of perfect graphs. In T. Ibaraki, T. Inagaki, K. Iwama, T. Nishizaki, and M. Yamashita, editors, *Proceedings of the 3rd International Symposium on Algorithms and Computation*, volume 650 of *Lecture Notes in Computer Science, ISAAC'92*, pages 116–125. Springer, 1992.

[78] T. Kloks, H. L. Bodlaender, H. Müller, and D. Kratsch. Computing treewidth and minimum fill-in: All you need are the minimal separators. In T. Lengauer, editor, *Proceedings of the 1st International Annual European Symposium on Algorithms*, volume 726 of *Lecture Notes in Computer Science, ESA'93*, pages 260–271. Springer, 1993.

[79] Y. Kobayashi, H. Maruta, Y. Nakae, and H. Tamaki. A linear edge kernel for two-layer crossing minimization. In D.-Z. Du and G. Zhang, editors, *Proceedings of the 19th Annual International Computing and Combinatorics Conference*, volume 7936 of *Lecture Notes in Computer Science, COCOON'13*, pages 458–468. Springer, 2013.

[80] Y. Kobayashi and H. Tamaki. A fast and simple subexponential fixed parameter algorithm for one-sided crossing minimization. In L. Epstein and P. Ferragina, editors, *Proceedings of the 20th Annual European Conference on Algorithms*, volume 7501 of *Lecture Notes in Computer Science, ESA'12*, pages 683–694. Springer, 2012.

[81] M. Koivisto and P. Parviainen. A space-time tradeoff for permutation problems. In *Proceedings of the 21st Annual ACM-SIAM Symposium on Discrete Algorithms*, SODA'10, pages 484–492. SIAM, 2010.

[82] M. Lampis, G. Kaouri, and V. Mitsou. On the algorithmic effectiveness of digraph decompositions and complexity measures. In S.-H. Hong, H. Nagamochi, and T. Fukunaga, editors, *Proceedings of the 19th International Symposium on Algorithms and Computation*, volume 5369 of *Lecture Notes in Computer Science, ISAAC'08*, pages 220–231. Springer, 2008.

[83] T. Lengauer. *Combinatorial algorithms for integrated circuit layout.* Wiley, 1990.

[84] D. Lokshtanov, D. Marx, and S. Saurabh. Lower bounds based on the exponential time hypothesis. *Bulletin of the EATCS*, 105:41–72, 2001.

[85] Z. Michalewicz and D. B. Fogel. *How to solve it: modern heuristics.* Springer, 2004.

[86] B. Monien and I. H. Sudborough. Min cut is np-complete for edge weighted trees. *Theoretical Computer Science*, 58(1–3):209–229, 1988.

[87] J. W. Moon and L. Moser. On cliques in graphs. *Israel Journal of Mathematics*, 3:23–28, 1965.

[88] R. Motwani and P. Raghavan. *Randomized algorithms.* Cambridge University Press, 1995.

[89] X. Muñoz, W. Unger, and I. Vrt'o. One sided crossing minimization is np-hard for sparse graphs. In M. Jünger and S. Leipert, editors, *Proceedings of the 9th International Symposium on Graph Drawing*, volume 2265 of *Lecture Notes in Computer Science, GD'01*, pages 115–123. Springer, 2002.

[90] H. Nagamochi. An improved bound on the one-sided minimum crossing number in two-layered drawings. *Discrete and Computational Geometry*, 33(4):569–591, 2005.

[91] H. Nagamochi. On the one-sided crossing minimization in a bipartite graph with large degree. *Theoretical Computer Science*, 332:417–446, 2005.

[92] H. Nagamochi. Linear layouts in submodular systems. In K.-M. Chao, T.-S. Hsu, and D.-T. Lee, editors, *Proceedings of the 23rd International Symposium on Algorithms and Computation*, volume 7676 of *Lecture Notes in Computer Science, ISAAC'12*, pages 475–484. Springer, 2012.

[93] M. Newton, O. Sýkora, and I. Vrt'o. Two new heuristics for two-sided bipartite graph drawing. In M. T. Goodrich and S. G. Kobourov, editors, *Proceedings of the 10th International Symposium on Graph Drawing*, volume 2528 of *Lecture Notes in Computer Science, GD'02*, pages 312–319. Springer, 2002.

[94] R. Niedermeier. *Invitation to fixed-parameter algorithms.* Oxford university press, 2006.

[95] A. Prokurowski and J. A. Telle. Classes of graphs with restricted interval models. *Discrete Mathematics and Theoretical Computer Science*, 3:167–176, 1999.

[96] H. Purchase. Which aesthetic has the greatest effect on human understanding? In G. D. Battista, editor, *Proceedings of the 5th International Symposium on Graph Drawing*, volume 1353 of *Lecture Notes in Computer Science, GD'97*, pages 248–261. Springer, 1997.

[97] N. Robertson and P. D. Seymour. Graph minors. i. excluding a forest. *Journal of Combinatorial Theory, Series B*, 35(1):39–61, Aug. 1983.

[98] N. Robertson and P. D. Seymour. Graph minors. viii. the disjoint path problem. *Journal of Combinatorial Theory, Series B*, 63(1):65–110, 1995.

[99] M. Schaefer. The graph crossing number and its variants: a survey. *The Electronic Journal of Combinatorics*, 20(2), 2013.

[100] P. D. Seymour and R. Thomas. Graph searching and a min-max theorem for tree-width. *Journal of Combinatorial Theory, Series B*, 58(1):22–33, 1993.

[101] F. Shahrokhi, O. Sýkora, L. A. Székely, and I. Vrt'o. On bipartite drawings and the linear arrangement. *SIAM Journal on Computing*, 30(6):1773–1789, 2001.

[102] F. Solano and M. Pióro. Lightpath reconfiguration in WDM networks. *IEEE/OSA Journal of Optical Communications and Networking*, 2(12), 2010.

[103] J. Spinrad, A. Brandstädt, and L. Stewart. Bipartite permutation graphs. *Discrete Applied Mathematics*, 18(3):279–292, 1987.

[104] K. Suchan and I. Todinca. Pathwidth of circular-arc graphs. In A. Brandstädt, D. Kratsch, and H. Müller, editors, *Proceedings of the 33rd International Workshop on Graph-Theoretic Concepts in Computer Science*, volume 4769 of *Lecture Notes in Computer Science, WG'07*, pages 258–269. Springer, 2007.

[105] K. Suchan and Y. Villanger. Computing pathwidth faster than 2^n. In J. Chen and F. V. Fomin, editors, *Proceedings of the 4th International Workshop on Parameterized and Exact Computation*, volume 5917 of *Lecture Notes in Computer Science, IWPEC'09*, pages 324–335. Springer, 2009.

[106] M. Suderman. Pathwidth and layered drawing of trees. *International Journal of Computational Geometry and Applications*, 14(3):203–225, 2004.

[107] M. Suderman. *Layered graph drawing*. PhD thesis, School of Computer Science McGill University, 2005.

[108] K. Sugiyama, S. Tagawa, and M. Toda. Methods for visual understanding of hierarchical system structures. *IEEE Transactions on Systems, Man and Cyvernetics*, 11(2):109–125, 1981.

[109] H. Tamaki. A polynomial time algorithm for bounded directed pathwidth. In P. Kolman and J. Ktatochvíl, editors, *Proceedings of the 37th International Workshop on Graph-Theoretic Concepts in Computer Science*, volume 6986 of *Lecture Notes in Computer Science, WG'11*, pages 331–342. Springer, 2011.

[110] V. Valls, R. Marti, and P. Lino. A branch and bound algorithm for minimizing the number of crossing arcs in bipartite graphs. *European Journal of Operational Research*, 90(2):303–319, 1996.

[111] M. S. Waterman and J. R. Griggs. Interval graphs and maps of dna. *Bulletin of Mathematical Biology*, 48(2):189–195, 1986.

[112] D. P. Williamson and D. B. Shmoys. *The design of approximation algorithms*. Cambridge University Press, 2011.

[113] G. J. Woeginger. Exact algorithms for np-hard problems: a survey. In M. Jünger, G. Reinelt, and G. Rinaldi, editors, *Combinatorial Optimization – Eureka, You Shrink!*, volume 2570 of *Lecture Notes in Computer Science*, pages 185–207. Springer, 2003.

[114] B. Yang and Y. Cao. Digraph searching, directed vertex separation and directed pathwidth. *Discrete Applied Mathematics*, 156(10):1822–1837, 2008.

[115] 中江勇介. ２部グラフの２階層描画の辺交差数最小化. Master's thesis, 明治大学大学院理工学研究科, 2012.