

Subtrees Search, Cycle Spectra and Edge-Connectivity Structures

by
On-Hei Solomon Lo

M.Sc., Rheinische Friedrich-Wilhelms-Universität Bonn, 2015
B.Sc., The Chinese University of Hong Kong, 2013

Thesis Submitted in Partial Fulfillment of the
Requirements for the Degree of
Doctor rerum naturalium

in the
Institut für Mathematik
Fakultät für Mathematik und Naturwissenschaften
Technische Universität Ilmenau

Supervisor: Jun.-Prof. Dr. Jens M. Schmidt
Referees: Prof. Dr. Matthias Kriesell
Ass. Prof. Dr. Tomáš Madaras

Date of Submission: 19th March 2019
Date of Defense: 16th September 2019

© 2019 by On-Hei Solomon Lo. All rights reserved.

ABSTRACT. In the first part of this thesis, we study subtrees of specified weight in a tree T with vertex weights $c : V(T) \rightarrow \mathbb{N}$. We introduce an overload-discharge method, and discover that there always exists some subtree S whose weight $c(S) := \sum_{v \in V(S)} c(v)$ is close to $\frac{c(T)}{2}$; the smaller the weight $c(T)$ of T is, the smaller difference between $c(S)$ and $\frac{c(T)}{2}$ we can assure. We also show that such a subtree can be found efficiently, namely in linear time. With this tool we prove that every planar hamiltonian graph $G = (V(G), E(G))$ with minimum degree $\delta \geq 4$ has a cycle of length k for every $k \in \{\lfloor \frac{|V(G)|}{2} \rfloor, \dots, \lceil \frac{|V(G)|}{2} \rceil + 3\}$ with $3 \leq k \leq |V(G)|$. Such a cycle can be found in linear time if a Hamilton cycle of the graph is given.

In the second part of the thesis, we present three cut trees of a graph, each of them giving insights into the edge-connectivity structure. All three cut trees have in common that they cover a given binary symmetric irreflexive relation on the vertex set of the graph, while generalizing Gomory-Hu trees. With these cut trees we show the following: (i) every simple graph G with $\delta \geq 5$ or with edge-connectivity $\lambda \geq 4$ or with vertex-connectivity $\kappa \geq 3$ contains at least $\frac{1}{24}\delta|V(G)|$ pendant pairs, where a pair of vertices $\{v, w\}$ is pendant if $\lambda_G(v, w) = \min\{d_G(v), d_G(w)\}$; (ii) every simple graph G satisfying $\delta > 0$ has $O(|V(G)|/\delta)$ δ -edge-connected components, and there are only $O(|V(G)|)$ edges left if these components are contracted; (iii) given a simple graph G satisfying $\delta > 0$, one can find some vertex subsets in near-linear time such that all non-trivial min-cuts are preserved, and $O(|V(G)|/\delta)$ vertices and $O(|V(G)|)$ edges remain when these vertex subsets are contracted.

ZUSAMMENFASSUNG. Im ersten Teil dieser Dissertation untersuchen wir Teilbäume eines Baumes T mit vorgegebenen Knotengewichten $c : V(T) \rightarrow \mathbb{N}$. Wir führen eine Overload-Discharge-Methode ein, und zeigen, dass es immer einen Teilbaum S gibt, dessen Gewicht $c(S) := \sum_{v \in V(S)} c(v)$ nahe $\frac{c(T)}{2}$ liegt. Je kleiner das Gewicht $c(T)$ von T ist, desto geringer ist dabei die Differenz zwischen $c(S)$ und $\frac{c(T)}{2}$, die wir sicherstellen können. Wir zeigen auch, dass ein solcher Teilbaum effizient, nämlich in Linearzeit, berechnet werden kann. Unter Ausnutzung dieser Methode beweisen wir, dass jeder planare hamiltonsche Graph $G = (V(G), E(G))$ mit Mindestgrad $\delta \geq 4$ einen Kreis der Länge k für jedes $k \in \{\lfloor \frac{|V(G)|}{2} \rfloor, \dots, \lceil \frac{|V(G)|}{2} \rceil + 3\}$ mit $3 \leq k \leq |V(G)|$ enthält. Dieser kann in Linearzeit berechnet werden, falls ein Hamilton-Kreis des Graphen bekannt ist.

Im zweiten Teil der Dissertation stellen wir drei Schnittbäume eines Graphen vor, von denen jeder Einblick in die Kantenzusammenhangsstruktur des Graphen gibt. Allen drei Schnittbäumen ist gemeinsam, dass sie eine bestimmte binäre symmetrische irreflexive Relation auf der Knotenmenge des Graphen überdecken; die Bäume können als Verallgemeinerungen von Gomory-Hu-Bäumen aufgefasst werden. Die Schnittbäume implizieren folgende Aussagen: (i) Jeder schlichte Graph G , der $\delta \geq 5$ oder Kantenzusammenhang $\lambda \geq 4$ oder Knotenzusammenhang $\kappa \geq 3$ erfüllt, enthält mindestens $\frac{1}{24}\delta|V(G)|$ zusammengehörige Paare, wobei ein Paar von Knoten $\{v, w\}$ zusammengehörig ist, falls $\lambda_G(v, w) = \min\{d_G(v), d_G(w)\}$ ist. (ii) Jeder schlichte Graph G mit $\delta > 0$ hat $O(|V(G)|/\delta)$ δ -kantenzusammenhängende Komponenten, und es verbleiben lediglich $O(|V(G)|)$ Kanten, wenn diese Komponenten kontrahiert werden. (iii) Für jeden schlichten Graphen G mit $\delta > 0$ sind Knotenmengen derart effizient berechenbar, dass alle nicht trivialen minimalen Schnitte erhalten bleiben, und $O(|V(G)|/\delta)$ Knoten und $O(|V(G)|)$ Kanten verbleiben, wenn diese Knotenmengen kontrahiert werden.

In memory of my grandma.

Preface

In this thesis we study two topics in structural and algorithmic graph theory. We are particularly interested in the notion of edge cut and edge-connectivity in graphs. It arises frequently in issues of network design and network reliability, and has been extensively studied for many decades. In the first part of the thesis, we introduce a method which allows us to find cycles of specified length in planar hamiltonian graphs under certain conditions. Note that cycles in a plane graph is known to be equivalent to edge cuts in the dual of the plane graph. We develop our tool in Chapter 1 and illustrate its applications in detail in Chapter 2. In the second part, comprised of joint works with Jens M. Schmidt and Mikkel Thorup, we introduce a structure called cut tree. While generalizing the classic Gomory-Hu tree, results concerning edge-connectivity can be derived from various cut trees. Mainly, it reveals how small edge cuts are located in a graph; details will be given in Chapter 3.

Another issue that concerns us most is how efficient we can obtain these graph-theoretical objects. In an era of *Big Data* the demand of linear or even sublinear algorithms now becomes much greater than ever. We complement our results with several efficient algorithms, for instance, a linear time algorithm for finding cycles of specified length and a near-linear time contraction-based sparsification preserving small edge cuts.

Acknowledgments. I would like to thank everybody who has offered me any kind of help during these years.

First and foremost, I would like to extend my gratitude to Jens M. Schmidt, my advisor, mentor and collaborator, from whom I have learnt much in every aspect. This thesis would not have been possible without his advice, guidance and ideas. It was also a great pleasure of mine to join his project funded by the German Research Foundation, and the Ilmenau-Košice graph theory workshop co-funded by the German Academic Exchange Service and the Ministry of Education, Science, Research and Sport of the Slovak Republic.

I am grateful to Samuel Mohr, who not only shared and elaborated his problem and idea to me, but also reviewed my manuscript and gave me many invaluable comments. I am much indebted to Thomas Böhme for his effort to secure the funding for the prolongation of my study. Besides, I am thankful to Rita Gerlach, Jochen Harant, Matthias Kriesell, Tomáš Madaras, Anja Pruchnewski, Thomas Schweser, and Michael Stiebitz for their substantial support in my research and teaching.

My deepest thanks to my family for their continuous faith in me.

Ilmenau, Spring 2019

Solomon Lo

Notation

In this section we mention the notation that shall be standard throughout this thesis.

Sets. We use the standard set-theoretical notation while we mention the following. $|X|$ denotes the *size* of a set X . $A - B$ represents the set $\{x : x \in A \text{ and } x \notin B\}$; parentheses can be omitted if B is a *singleton*, i.e. $|B| = 1$. Given a set A , a *binary relation* R on A is a subset of $\{(a, a') : a, a' \in A\}$. R is *symmetric* if $(a, a') \in R$ implies $(a', a) \in R$ for all $a, a' \in A$; *irreflexive* if $(a, a) \notin R$ for all $a \in A$. It is natural to see elements of a symmetric irreflexive binary relation as unordered pairs.

Arithmetic. We recall some standard arithmetic notation. For $x \in \mathbb{R}$, $|x|$ denotes the absolute value of x , $\lceil x \rceil$ denotes the smallest integer not less than x , and $\lfloor x \rfloor$ denotes the greatest integer not exceeding x . For two functions $f : \mathbb{N} \rightarrow \mathbb{R}$ and $g : \mathbb{N} \rightarrow \mathbb{R}$, we write $f = O(g)$ if there exist $N_0 \in \mathbb{N}$ and $c \in \mathbb{R}$ such that $|f(n)| \leq c|g(n)|$ for all $n \geq N_0$; $f = \Omega(g)$ if $g = O(f)$; and $f = o(g)$ if $\frac{f(n)}{g(n)}$ tends to 0 when n tends to infinity.

Graphs.

Undirected Graphs. An *undirected graph* is a pair $G = (V(G), E(G))$ consisting of a set $V(G)$, whose members are called *vertices*, and $E(G)$ a subset of $\{\{u, v\} : u, v \in V(G) \text{ and } u \neq v\}$, whose members are called *edges*. A vertex v is *incident* with an edge e if $v \in e$. The two vertices incident with an edge are its *endvertices*. Two vertices u, v of G are *adjacent*, or *neighbors*, if $\{u, v\} \in E(G)$. For a vertex $v \in V(G)$, let $N_G(v)$ be the set of neighbors of v in G .

Given a vertex set $W \subseteq V(G)$, $G - W$ denotes the graph on the vertex set $V(G) - W$ with edges both of whose endvertices are in W . Given an edge set $F \subseteq E(G)$, $G - F$ denotes the graph on the vertex set $V(G)$ with edge set $E(G) - F$. A *subgraph* is obtained by deleting edges and vertices. A *spanning subgraph* is obtained by deleting edges only. $G[W]$ is defined to be the *induced subgraph* of G on W , i.e. $G - (V(G) - W)$.

For non-empty and disjoint vertex subsets $X, Y \subset V(G)$, let $E_G(X, Y)$ denote the set of all edges in G that have one endvertex in X and one endvertex in Y . Let further $\bar{X} := V(G) - X$, $d_G(X, Y) := |E_G(X, Y)|$, and $d_G(X) := |E_G(X, \bar{X})|$; if $X = \{v\}$ for some vertex $v \in V(G)$, we simply write $E_G(v, Y)$, $d_G(v, Y)$ and $d_G(v)$. The number $d_G(v)$ is also called the *degree* of v in G . In order to increase readability, we will omit subscripts whenever it is clear from the context. We define $\delta(G)$ or simply δ to be the *minimum degree* $\min_{v \in V(G)} d_G(v)$ of G .

We also denote by uv or vu the edge with endvertices $u, v \in V(G)$. We abuse the notation of a sequence of vertices as follows. For t distinct vertices v_1, v_2, \dots, v_t , we denote by $v_1v_2 \dots v_t$ the *path* P of length $t - 1$ and size t with *endvertices* v_1, v_t such that $V(P) := \{v_1, v_2, \dots, v_t\}$ and $E(P) := \{v_1v_2, v_2v_3, \dots, v_{t-1}v_t\}$. For $t \geq 3$ distinct vertices v_1, v_2, \dots, v_t , we denote by $v_1v_2 \dots v_tv_1$ the *cycle* K of length t such that $V(K) := \{v_1, v_2, \dots, v_t\}$ and $E(K) := \{v_1v_2, v_2v_3, \dots, v_{t-1}v_t, v_tv_1\}$. A *connected component* in a graph is a maximal subgraph H

such that there is a path in G between any two vertices of $V(H)$. A graph is *connected* if it consists of one component. A *forest* is a graph without cycles. A *tree* is a connected graph without cycles. A *subtree* is a connected subgraph of a tree. Let T be a tree. For $vw \in E(T)$, we denote by $T[vw; v]$ the connected component of $T - vw$ containing v . Given a vertex $a \in V(T)$, we specify the tree T rooted at a by $T^{(a)}$. For $v \in V(T)$, $T_v^{(a)}$ is defined as the subtree of T containing v and all of its descendants in $T^{(a)}$. In other words, let P be the path with endvertices a, v , $T_v^{(a)}$ is the connected component containing v in $T - E(P)$. Vertices of degree 1 in a tree are called *leaves*.

A subset $\emptyset \neq X \subset V$ of a graph G is called a *cut* of G . We say that a cut X *separates* $u, v \in V(G)$ if exactly one of u, v is in X ; we also call X a *u - v -cut*. We define by $d_G(X)$ the *size* of the cut X . A cut X of G is *trivial* if $|X| = 1$ or $|\bar{X}| = 1$. For two vertices $u, v \in V(G)$, let $\lambda_G(u, v)$ be the maximal number of edge-disjoint paths between u, v in G . A *minimum u - v -cut* is a cut X that separates u and v and satisfies $d_G(X) = \lambda_G(u, v)$. Two vertices $u, v \in V(G)$ are called *k -edge-connected* if $\lambda_G(u, v) \geq k$. For any k , the *k -edge-connectedness* relation on vertices is symmetric and transitive, and thus its reflexive closure is an equivalence relation that partitions $V(G)$; let the *k -edge-connected components* be the blocks of this partition. The *edge-connectivity* $\lambda(G)$ or simply λ of G is the greatest integer such that every two distinct vertices are $\lambda(G)$ -edge-connected if $|V(G)| \geq 2$, and it is infinity if $|V(G)| = 1$. A cut X is a *min-cut* if $d_G(X) = \lambda(G)$. The *vertex-connectivity* $\kappa(G)$ or simply κ of G is defined to be $|V(G)| - 1$ if G is a *complete* graph (in which every two vertices are adjacent) and otherwise the smallest integer $\kappa(G)$ such that there exists a vertex subset $S \subseteq V(G)$ of size $\kappa(G)$ and $G - S$ is disconnected. G is *k -connected* if $\kappa(G) \geq k$.

A *planar graph* is a graph that can be embedded in the Euclidean plane \mathbb{R}^2 , i.e. it can be drawn on the plane in such a way that its edges intersect only at their endvertices but do not cross each other. Such a drawing is called a *planar embedding* of the graph. A *plane graph* is a planar graph with some fixed planar embedding. For a plane graph G , we call the open regions in $\mathbb{R}^2 - (V(G) \cup E(G))$ *faces*. We consider only 2-connected planar graphs. In this case the boundaries of the faces are cycles of the graph. Two faces are *adjacent* if their boundaries intersect. The *dual graph* G^* of G is the graph whose vertex set is the set of faces of G and two vertices of G^* are adjacent if the two corresponding faces are adjacent. We identify the faces of G not only with the vertices in the dual graph G^* but also with the cycles in the boundaries of the faces provided that G is not a cycle.

Directed Graphs. A *directed graph* $G = (V(G), E(G))$ is a pair consisting of a vertex set $V(G)$, and an edge set $E(G)$ which is a subset of $\{(u, v) : u, v \in V(G) \text{ and } u \neq v\}$. We also call $(u, v) \in E(G)$ a *directed edge* or an edge directed from u to v ; we denote it also by uv . *Directed paths* and *directed cycles* can be defined analogously. Let C be a directed cycle. For $u, v \in V(C)$, we define $[u, v]_C$ to be the path directed from u to v along C . Subscripts can be omitted if it is clear from the context. Let uv be an edge in C , we define $u^+ := v$ and $v^- := u$.

Multigraphs and Simple Graphs. An undirected or directed *multigraph* is a graph whose edge set is a multiset, i.e. a set in which a multiple of occurrences of the same element is allowed. We call a graph *simple* if every edge has only one occurrence in the edge set. We consider only finite and simple graphs, unless otherwise specified. Given a graph or multigraph G , *contracting* a vertex subset $X \subseteq V(G)$ means identifying all vertices in X and deleting the occurring self-loops (we do not require that X induces a connected graph in G). Note that contracting a vertex subset in a simple graph can possibly give a multigraph.

Contents

Preface	vi
Notation	vii
Part I.	1
Chapter 1. Subtrees of Specified Weight	3
1.1. Introduction	3
1.2. An Overload-Discharge Approach	4
1.3. Support Vertices and Support Subtree	7
1.4. Overloading Vertices and Discharges	8
1.5. Proof of the Lemma 1.1	10
1.6. Some Examples	12
Chapter 2. Cycles of Specified Length	13
2.1. Overview of Cycle Spectra of Planar Graphs	13
2.2. Cycles of Length Close to Medium-Length	14
2.3. Planar Hamiltonian Graphs	15
2.4. 3-Connected Planar Hamiltonian Graphs	16
Part II.	19
Chapter 3. Generalized Cut Trees for Edge-Connectivity	21
3.1. Introduction	21
3.2. Cut Trees	22
3.3. Pendant Trees and Pendant Pairs	23
3.4. Contraction-Based Sparsification Preserving Small Cuts	30
Bibliography	36
Symbol Index	38
Index	39

Part I

CHAPTER 1

Subtrees of Specified Weight

1.1. Introduction

Trees, as one of the most fundamental classes of graphs, have a long history for being used to model many problems. They have numerous applications, for example, in searching, data storage and network design. We can assign weights to the vertices in a tree and ask, for example, how to find a subtree of maximum or minimum weight, or we can ask how to partition the tree into subtrees satisfying some certain requirements. We refer the reader to [Gar05, AMLM13] for optimization problems regarding weighted trees, and to [Vyg11, INS⁺12] for tree partitioning problems. In this chapter we restrict the vertex weights to positive integers, and we ask whether there is a subtree whose weight is very close to half of the weight of the whole tree.

Given a tree T and vertex weights $c : V(T) \rightarrow \mathbb{N}$, it is natural to ask subtrees of which specified weight would exist. Let S be a subtree of T . We define $c(S) := \sum_{v \in V(S)} c(v)$. Let $k, g \in \mathbb{N}$ with $1 \leq k \leq c(T)$. We aim at finding a subtree S of weight $k - g + 1 \leq c(S) \leq k$. Denote by N_1 and N_2 the number of vertices of the tree T and the weight $c(T)$ of the whole tree, respectively. Note that if we allow N_2 to be arbitrarily large when compared to N_1 , then it would be hopeless for us to achieve our goal. For example, it can happen that every vertex has weight, say $g' \gg g$, then a subtree S of weight $k - g + 1 \leq c(S) \leq k$ exists if and only if $k \equiv 0, \dots, g - 1 \pmod{g'}$. It means that the desired subtree does not exist for most choices of k . We describe by h the difference between N_1 and $\frac{N_2}{2}$. Our main goal is to prove the following lemma, which can be interpreted as that the closer the value of our target k to the medium-weight $\frac{N_2}{2}$ and the smaller the medium-weight $\frac{N_2}{2}$ when compared to the number of vertices N_1 , the more favourable conditions we have in finding the desired subtree. It is complemented by a deterministic linear-time algorithm, which will be given in Section 1.5. We will find applications of this lemma in Chapter 2.

LEMMA 1.1. *Let $k, g, N_1, N_2 \in \mathbb{N}$ such that $1 \leq k \leq N_2$, $2g + h > 3$ and $2k - 4g - h + 3 \leq N_2 \leq 2k + g + h - 2$, where $h := 2N_1 - N_2$. Let T be a tree of N_1 vertices and let $c : V(T) \rightarrow \mathbb{N}$ be vertex weights such that $c(T) = N_2$ and $c(v) \leq k$ for all $v \in V(T)$. Then there exists a subtree S of T of weight $k - g + 1 \leq c(S) \leq k$ and S can be found in $O(N_1)$ time.*

For the running time we assume that each arithmetic operation can be done in constant time.

Along with the existence of subtrees of specified weight, we present an optimal linear-time algorithm for finding them. Note that a tree may have exponentially many subtrees in general. Hence, in our algorithm only a rather restricted (linear-size) subclass of subtrees will be considered. We will exploit the Euler tour technique and find a subtree by local search. Formally, given a tree $T = (V(T), E(T))$, we construct a directed cycle C_T of size $2|E(T)|$ and consider the canonical homomorphism which maps vertices of C_T to that of T (as depicted in Figure 1.1). It is clear that every path in C_T will be correspondingly mapped

to a subtree of T . We will prove that it is indeed the case that there exists such a subtree satisfying the requirement. Therefore a linear-time algorithm follows as a simple consequence, which searches greedily for a path in C_T such that its corresponding subtree in T is what we are looking for. To prove that such a path in C_T exists, we assume it is not the case, and then deduce a contradiction by counting the number of vertices of weight 1 in T in two ways.

We organize as follows. We introduce the aforementioned restricted subclass of subtrees in Section 1.2.1. Since then we assume, towards a contradiction, that there is no subtree of T having the desired weight in this subclass. We outline how to prove Lemma 1.1 by double counting in Section 1.2.2. We study the subtrees in this linear-size subclass in Section 1.4, and give a proof of Lemma 1.1 and a pseudocode of a linear-time algorithm in Section 1.5. In Section 1.6 we present some examples showing that the conditions in Lemma 1.1 are tight from several aspects.

1.2. An Overload-Discharge Approach

1.2.1. Overloading Subtrees by ETT. To define the subclass of subtrees mentioned above, we consider the subtrees collected by the so-called *Euler tour technique* (ETT) which was first introduced by Tarjan and Vishkin [TY84] and has abundant applications in computing and data structures.

We assume a fixed planar embedding of the tree T and we walk around it, i.e. we see edges of T as walls perpendicular to the plane and we walk on the plane along the walls. This walk yields a cycle of size $2(|V(T)| - 1)$.

To make it precise, we define the auxiliary directed cycle graph C_T as follows. For each $v \in V(T)$, we enumerate the edges incident to v in the clockwise order according to the planar embedding and denote them by $e_{v,1}, e_{v,2}, \dots, e_{v,d_T(v)}$. The vertex set $V(C_T)$ consists of $d_T(v)$ vertices $w_{v,1}, w_{v,2}, \dots, w_{v,d_T(v)}$ for each $v \in V(T)$. Let $w_{v,d_T(v)+1} := w_{v,1}$. And, for every edge $uv \in E(T)$, say $uv = e_{u,i} = e_{v,j}$ for some $i \in \{1, \dots, d_T(u)\}$ and $j \in \{1, \dots, d_T(v)\}$, $E(C_T)$ contains the edges $w_{u,i}w_{v,j+1}$ and $w_{v,j}w_{u,i+1}$. It is clear that C_T is our desired cycle of size $2(|V(T)| - 1)$ (see Figure 1.1).

Note that a directed path in C_T can be naturally corresponded to a subtree in T . Moreover, growing a subtree by this walking-around-walls DFS in T is equivalent to growing a directed path in C_T .

We define the mapping ρ (a homomorphism) from $V(C_T)$ to $V(T)$ by $\rho(w_{v,i}) := v$ for $w_{v,i} \in V(C_T)$ with $v \in V(T)$ and $i \in \{1, \dots, d_T(v)\}$. We also extend this mapping for paths $[u, v]$ directed from u to v in C_T ($u, v \in V(C_T)$) by defining $\rho([u, v]) := T[\{\rho(w) : w \in V([u, v])\}]$. We then extend the weight function c to the vertices w and directed paths $[u, v]$ in C_T ($w, u, v \in V(C_T)$) by $c(w) := c(\rho(w))$ and $c([u, v]) := c(\rho([u, v]))$.

Here we state an assumption (towards a contradiction) which we adopt from now on:

Assumption (Ω). There are no $x, y \in V(C_T)$ with $k - g + 1 \leq c([x, y]) \leq k$.

In other words there is no subtree of T with weight between $k - g + 1$ and k can be found by searching along the Euler tour. Once we show that Assumption (Ω) cannot hold, we can assure that there exists a subtree S of T having weight $k - g + 1 \leq c(S) \leq k$. Indeed, we can have some linear-size subclass of subtrees that contains some subtree S having weight $k - g + 1 \leq c(S) \leq k$. For instance, if $k < c(T)$, we can consider the subtrees $\rho([u, v])$ ($u, v \in V(C_T)$) satisfying $c([u, v]) \leq k$, $c([u^-, v]) > k$ and $c([u, v^+]) > k$. There are at most $|V(C_T)| = O(|V(T)|)$ such subtrees, and at least one of them is of weight between $k - g + 1$

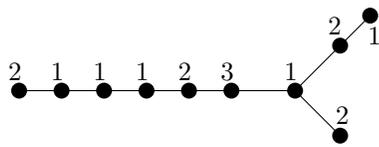
and k when Assumption (Ω) doesn't hold. It helps us to devise a linear-time algorithm (Algorithm 1) for searching a subtree of the desired weight.

By Assumption (Ω) , the inequalities $c([u, v]) \geq k - g + 1$ and $c([u, v]) \leq k$ ($u, v \in V(C_T)$) are equivalent to $c([u, v]) > k$ and $c([u, v]) < k - g + 1$, respectively. (Such usage of Assumption (Ω) would occur tacitly.)

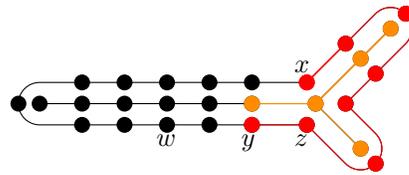
We mention some more consequences that follow from Assumption (Ω) . It is readily to see that $c(v) \leq k - g$ for any $v \in V(T)$ and $N_2 > k$. Consider a path $[u, v]$ ($u, v \in V(C_T)$) satisfying $c([u, v]) \geq k$, $c([u^+, v]) \leq k$ and $c([u, v^-]) \leq k$, equivalently, $c([u, v]) > k$, $c([u^+, v]) < k - g + 1$ and $c([u, v^-]) < k - g + 1$. It is clear that such a path exists, $\rho(u) \neq \rho(v)$, and both $c(u)$, $c(v)$ are at least $g + 1$, i.e. T has at least two distinct vertices of weight at least $g + 1$.

Let $u, v \in V(C_T)$. $[u, v]$ is k -overloading or simply *overloading* if $c([u^-, v^-]) > k$, $c([u, v^-]) \leq k$ and $c([u, v]) > k$, and we say $\rho([u, v])$ is an *overloading subtree*. Note that an overloading path always exists when we assume (Ω) , since we are given that $1 \leq k \leq N_2 = c(T)$ and $c(v) \leq k$ for every $v \in V(T)$.

1.2.2. Bounds on $|V_1|$. Now we see how we can count the number of vertices of weight 1 in two ways so that a contradiction may occur. For $i \in \mathbb{N}$, we denote by $V_i(T) \subseteq V(T)$ the set of vertices v in T with $c(v) = i$. We write $V_i := V_i(T)$ if there is no ambiguity. By



(a) The tree T with vertex weights.



(b) The tree T , and the auxiliary cycle C_T whose edges are directed clockwise. Set $k := 7$ and $g := 1$. $Q_{x,y}$ is a maximal overload-discharge quadruples, since $c([x, y]) > 7$, $c([x, y^-]) < 7$ and $c([x^-, y^-]) > 7$.

Figure 1.1. Walk around the tree by ETT.

considering the sum of all the vertex weights, we have that

$$\begin{aligned}
N_2 &= \sum_{i \geq 1} i|V_i| \\
&= 2 \sum_{i \geq 1} |V_i| + \sum_{1 \leq i \leq g} (i-2)|V_i| + \sum_{i \geq g+1} (i-2)|V_i| \\
&= 2N_1 - |V_1| + \sum_{2 \leq i \leq g} (i-2)|V_i| + \sum_{i \geq g+1} (i-g-1)|V_i| + \sum_{i \geq g+1} (g-1)|V_i| \\
&= \sum_{i \geq g+1} (i-g-1)|V_i| + \sum_{i \geq 2} \min\{i-2, g-1\}|V_i| + 2N_1 - |V_1|.
\end{aligned}$$

As $2N_1 - N_2 =: h$ and $\sum_{i \geq g+1} |V_i| \geq 2$, we have the following lower bound on $|V_1|$:

$$(\diamond) \quad |V_1| \geq \sum_{i \geq g+1} (i-g-1)|V_i| + 2g + h - 2.$$

The intuitive idea of our proof of Lemma 1.1 is that if $|V_1|$ is large enough, i.e. there are many vertices of weight 1, then it should facilitate the search of subtree of the desired weight. Therefore, by assuming that Lemma 1.1 does not hold, there would be some upper bound on the number of vertices of weight 1 which shows that the inequality (\diamond) must be contradicted. The upper bound is realized by the following observation.

OBSERVATION 1.2. *Let $g, k \in \mathbb{N}$. Let S be a subtree of T with $c(S) > k$, l be a leaf of S with $c(S-l) < k-g+1$, M be a subset of $V(S) - l$ and n a vertex in $S - M - l$ such that $S - M$ remains as a tree, n is a leaf of $S - M$, $c(S - M) > k$ and $c(S - M - n) < k - g + 1$. Then we have*

$$(*) \quad |M \cap V_1| \leq c(S) - (k+1) \leq c(l) - g - 1.$$

The vertex set M can be seen as a set of vertices which are collected from a leave-cutting process, i.e. we cut leaves (other than l) one by one from S with that the weight of the remainder still larger than k , and it becomes less than $k-g+1$ once we further cut the vertex n . Note that l is not cut from the subtree and it always stays as a leaf in the remaining part.

PROOF. Since $c(S) - c(M) = c(S - M) > k$ and $c(S) - c(l) = c(S - l) < k - g + 1$, we have

$$|M \cap V_1| \leq c(M) \leq c(S) - (k+1) \leq c(l) - g - 1.$$

□

Note that the conditions given in Observation 1.2 appear naturally if we have a tree T which has no subtree of weight between $k-g+1$ and k . We carry out an *overload-discharge process* as follows. We grow a subtree (say a single vertex) which is of weight less than $k-g+1$ until we grow it with a vertex l which makes the weight of the subtree at least $k-g+1$. As we assume that no subtree is of weight between $k-g+1$ and k , when we halt the growth, the weight of the subtree is actually not only at least $k-g+1$ but greater than k . We then start to cut its leaves (other than l) one by one until the weight declines to be less than $k-g+1$ again. The overload and discharge steps can always be achieved provided that $N_2 > k$ and $c(v) \leq k-g$ for all $v \in V(T)$. We say that l *overloads* S and a *discharge* $M \cup \{n\}$ containing the *last discharge* n follows, and that (S, l, M, n) is an *overload-discharge quadruple*. It is clear that l and n must have weight at least $g+1$.

Let us have a look of a crude argument on how a contradiction would occur. Suppose we have a family of overload-discharge quadruples (S_f, l_f, M_f, n_f) (with some indices f) such that the vertices of weight 1 in T is *covered* by the discharges, i.e. $V_1 \subseteq \bigcup_f M_f$, and each overloading vertex l_f corresponds to only one overload-discharge quadruple, then, by the inequality (*), we can simply deduce the following contradiction to the inequality (\diamond):

$$|V_1| \leq \sum_f |M_f \cap V_1| \leq \sum_f (c(l_f) - g - 1) \leq \sum_{i \geq g+1} (i - g - 1) |V_i|.$$

Although it is not always possible to have such a family of quadruples, we are still able to have some *sufficiently good* family which leads to a contradiction to the inequality (\diamond). We will consider the family of overload-discharge quadruples corresponding to overloading subtrees.

We demonstrate how an overload-discharge quadruple can be formed by considering paths in C_T . Let u, v be two distinct vertices of C_T . If $c([u, v^-]) < k - g + 1$ but $c([u, v]) > k$, then there exists $w \in V([u, v^-])$ such that $c([w, v]) > k$ and $c([w^+, v]) < k - g + 1$. It is clear that $\rho(v)$ overloads the subtree $\rho([u, v])$ and we call

$$(\rho([u, v]), \rho(v), V(\rho([u, v])) - V(\rho([w, v])), \rho(w)) =: Q_{u,v}$$

an *overload-discharge quadruple associated with u, v* .

An overload-discharge quadruple $Q_{u,v}$ associated with $u, v \in V(C_T)$ is *maximal* if $c([u^-, v^-]) > k$ holds, or equivalently, $[u, v]$ is an overloading path in C_T (see Figure 1.1(b)). We let $\mathcal{Q}(T; c, k) =: \mathcal{Q}$ be the family of all maximal overload-discharge quadruples associated with some $u, v \in V(C_T)$.

1.3. Support Vertices and Support Subtree

In order to see how the overloading subtrees from \mathcal{Q} would be *packed* in the weighted tree T , we need to study its structure in more detail. We introduce the notion of support vertices and support subtree of the weighted tree T in this section.

We first fix an arbitrary vertex $a \in V(T)$ and consider the rooted tree $T^{(a)}$. Note that there always exists a vertex r such that $c(T_r^{(a)}) > k$ and $c(T_w^{(a)}) \leq k$ for all children w of r in $T^{(a)}$, as we assume $c(T^{(a)}) = N_2 > k$. We then take one such vertex r and consider the tree $T^{(r)}$ rooted at r . Let r_1, \dots, r_t ($t \in \mathbb{N}$) be the vertices each satisfies that $c(T_{r_i}^{(r)}) > k$ and $c(T_w^{(r)}) \leq k$ for all children w of r_i in $T^{(r)}$ ($i = 1, \dots, t$). We call r, r_1, \dots, r_t *support vertices* of T , and the minimal subtree T^* containing all support vertices *support subtree* of T .

Note that $T_w^{(r)}$ is exactly the same subtree as $T_w^{(a)}$ for every $w \in V(T_r^{(a)}) - r$, therefore $c(T_w^{(r)}) = c(T_w^{(a)}) \leq k$ and $r_i \notin V(T_r^{(a)}) - r$ for every $i = 1, \dots, t$. If there are two distinct support vertices r_{i_1}, r_{i_2} with $i_1, i_2 \in \{1, \dots, t\}$, we have that r_{i_1} is neither ancestor nor descendant of r_{i_2} in $T^{(r)}$, and, in particular, both r_{i_1}, r_{i_2} cannot be r . It is possible that r is one of the r_1, \dots, r_t ; it happens if and only if r is the only support vertex and $|V(T^*)| = 1$. We conclude that the leaves of T^* are exactly the support vertices if $|V(T^*)| > 1$, as T^* is the union of the paths P_i ($i = 1, \dots, t$), where P_i is the path in T with endvertices r and r_i .

It is clear that $T - E(T^*)$ is a forest of $|V(T^*)|$ subtrees. For $\tilde{r} \in V(T^*)$, we denote by $T^*[\tilde{r}]$ the maximal subtree of $T - E(T^*)$ containing \tilde{r} . If $|V(T^*)| > 1$, then the subtrees $T_r^{(a)}, T_{r_1}^{(r)}, \dots, T_{r_t}^{(r)}$ are exactly $T^*[r], T^*[r_1], \dots, T^*[r_t]$, and each of them has weight at least $k + 1$.

Let t^* be the number of support vertices. It is clear that $t^* = t = 1$ if $|V(T^*)| = 1$, and $t^* = t + 1$ if $|V(T^*)| > 1$. We claim that $N_2 \geq t^*(k + 1)$. It holds trivially if $|V(T^*)| = 1$. If

$|V(T^*)| > 1$, then we have t^* vertex-disjoint subtrees $T^*[r], T^*[r_1], \dots, T^*[r_t]$. Thus we have

$$N_2 = c(T) \geq c(T^*[r]) + c(T^*[r_1]) + \dots + c(T^*[r_t]) \geq t^*(k+1).$$

In particular, $N_2 \geq 2k+2$ if $|V(T^*)| > 1$.

We remark that for a fixed k the support tree T^* is uniquely defined if $|V(T^*)| > 1$, while it is not always uniquely defined if $|V(T^*)| = 1$, as it would depend on the initial root a . For ease of presentation we assume that some support tree is fixed throughout.

1.4. Overloading Vertices and Discharges

In this section we focus on the vertices which overload subtrees from \mathcal{Q} and see how many discharges they can carry each. We first show a sufficient condition for a vertex to be contained in some discharge from \mathcal{Q} .

LEMMA 1.3. *Let vw be an edge in T . If $c(T[vw; w]) \geq k-g$, then there exists $(S, l, M, n) \in \mathcal{Q}$ with $v \in M \cup \{n\}$.*

PROOF. Let $i \in \{1, \dots, d_T(v)\}$ such that the edge vw is $e_{v,i}$. We grow a path in C_T from $u := w_{v,i}$ to obtain an overload-discharge quadruple $Q_{u,y}$ associated with u, y for some $y \in V(C_T)$. The corresponding situation in T is that a subtree starts growing at v , then traverses along the edge $e_{v,i}$ immediately. It will overload, i.e. the weight reaches larger than k , without revisiting v , since $c(v) + c(T[vw; w]) \geq k-g+1$.

We can augment the path $[u, y]$ backwards along the cycle C_T to obtain $[x, y]$ such that $c([x, y^-]) < k-g+1$ but $c([x^-, y^-]) > k$. Then we have $Q_{x,y} \in \mathcal{Q}$. Note that u is the only vertex in $[u, y]$ with $\rho(u) = v$, and u cannot stay in the path after discharge since $c([u, y]) \geq k-g+1$. Therefore the discharge of $Q_{x,y}$ must contain v . \square

Now we give a necessary condition for a vertex to be an overloading vertex in some quadruple in \mathcal{Q} .

LEMMA 1.4. *Let $Q_{x,y} \in \mathcal{Q}$ be an overload-discharge quadruple associated with some $x, y \in V(C_T)$. We have $c(T[\rho(y)\rho(y^-); \rho(y^-)]) + c(\rho(y)) > k$.*

PROOF. It is clear that the subtree $\rho([x, y^-])$ is contained in the subtree $T[\rho(y)\rho(y^-); \rho(y^-)]$. Therefore, $c(T[\rho(y)\rho(y^-); \rho(y^-)]) + c(\rho(y)) \geq c([x, y]) > k$ as $\rho(y)$ overloads $\rho([x, y])$. \square

We next show that if there are more than one overloading subtrees having the same overloading vertex, then the mutual intersection among these subtrees can only be the overloading vertex, and such a vertex must be in the support subtree. We also prove upper bounds on these discharges.

LEMMA 1.5. *Let Q_{x_1, y_1} and Q_{x_2, y_2} be two distinct overload-discharge quadruples associated with x_1, y_1 and x_2, y_2 , where $x_1, y_1, x_2, y_2 \in V(C_T)$, in \mathcal{Q} , respectively. If $\rho(y_1) = \rho(y_2) =: l$, we have*

$$V(\rho([x_1, y_1])) \cap V(\rho([x_2, y_2])) = \{l\}$$

and l must be a vertex in the support subtree T^ .*

PROOF. As Q_{x_1, y_1} and Q_{x_2, y_2} are distinct overload-discharge quadruples, by the choice of maximality of elements of \mathcal{Q} , y_1 must be different from y_2 . Hence we have distinct indices $i, j \in \{1, \dots, d_T(l)\}$ such that $y_1 = w_{l, i+1}$ and $y_2 = w_{l, j+1}$. Note that y_f ($f = 1, 2$) is the only vertex in $[x_f, y_f]$ with $\rho(y_f) = l$ since l is the overloading vertex. It means that l is not in the subtree $\rho([x_f, y_f^-])$ ($f = 1, 2$). Moreover, the subtree $\rho([x_1, y_1^-])$ is a

subtree of $T[e_{l,i}; \rho(y_1^-)]$, i.e. the component not containing l when deleting the edge $e_{l,i}$, and similarly, $\rho([x_2, y_2^-])$ is a subtree of the component not containing l when deleting the edge $e_{l,j}$. Therefore $V(\rho([x_1, y_1^-])) \cap V(\rho([x_2, y_2^-])) = \emptyset$ and $V(\rho([x_1, y_1])) \cap V(\rho([x_2, y_2])) = \{l\}$.

Suppose $l \notin V(T^*)$. Let $u \in V(T^*)$ and w be the neighbor of l such that $u \in T[lw; w]$. By the definition of the support subtree, for every neighbor v of l other than w , we have $c(T[lv; v]) + c(v) \leq k$. By Lemma 1.4, there are at most one overloading subtree whose overloading vertex is l . \square

LEMMA 1.6. *Let T_0 be a subtree of T . Let l be an overloading vertex shared by $t \in \mathbb{N}$ quadruples $(S_i, l, M_i, n_i) \in \mathcal{Q}$, for $i = 1, \dots, t$, such that S_i is a subtree of T_0 for every $i = 1, \dots, t$. We have*

$$\sum_{i=1}^t |M_i \cap V_1| \leq c(l) + (t-2)(c(l) - k - 1) + c(T_0) - 2k - 2 \leq c(T_0) - k - 1.$$

PROOF. By Lemma 1.5, the overloading subtrees share only the overloading vertex l , hence $c(l) + \sum_i (c(S_i) - c(l)) \leq c(T_0)$. By Observation 1.2 and the assumption $c(v) \leq k$ for all $v \in V(T)$, we have $\sum_i |M_i \cap V_1| \leq \sum_i (c(S_i) - (k+1)) = \sum_i (c(S_i) - c(l)) + t(c(l) - (k+1)) \leq c(T_0) - c(l) + t(c(l) - (k+1)) = c(T_0) + c(l) - 2(k+1) + (t-2)(c(l) - k - 1) = c(T_0) - k - 1 + (t-1)(c(l) - k - 1) \leq c(T_0) - k - 1$. \square

As the last preparation for the proof of Lemma 1.1 we show that a reasonable portion of vertices will be covered by the discharges from \mathcal{Q} .

LEMMA 1.7. *If $|V(T^*)| > 1$, then we have*

$$\bigcup_{(S,l,M,n) \in \mathcal{Q}} (M \cup \{n\}) = V(T).$$

If $|V(T^)| = 1$ and $N_2 \geq 2k - 2g - D$ for some $D \in \mathbb{N}$, then we have $|\bigcup_{(S,l,M,n) \in \mathcal{Q}} (M \cup \{n\})| \geq |V(T)| - D$. If $|V(T^*)| = 1$ and $N_2 \geq 2k - 2g$, then we have*

$$\bigcup_{(S,l,M,n) \in \mathcal{Q}} (M \cup \{n\}) \supseteq V(T) - V(T^*).$$

PROOF. If $|V(T^*)| > 1$, for a vertex v in T , we can take a support vertex $u \neq v$ such that $v \notin T^*[u]$. Let w be the vertex adjacent to v such that $u \in T[vw; w]$. We have $c(T[vw; w]) \geq c(T^*[u]) \geq k+1 \geq k-g$, and hence, by Lemma 1.3, there exists $(S, l, M, n) \in \mathcal{Q}$ such that $v \in M \cup \{n\}$. Thus $\bigcup_{(S,l,M,n) \in \mathcal{Q}} (M \cup \{n\}) = V(T)$.

If $|V(T^*)| = 1$ and $N_2 \geq 2k - 2g - D$ for some $D \in \mathbb{N}$, let r be the only one support vertex. Consider the tree $T^{(r)}$ rooted at r . Let U be the set of vertices v with $c(T_v^{(r)}) \geq N_2 - k + g + 1$. For $v \in V(T) - r - U$, let w be the parent of v in $T^{(r)}$, we have $c(T[vw; w]) \geq N_2 - (N_2 - k + g) = k - g$ and hence, by Lemma 1.3, v is covered by some discharge from \mathcal{Q} . We can assume that U is not empty (otherwise at most one vertex, namely the root r , can be not covered by any discharge from \mathcal{Q}).

We consider the subtree $T[U]$ of T induced by U . If $T[U]$ has two leaves v, w other than r , then $|U| \leq 2 + c(T[U] - v - w) \leq 2 + N_2 - c(T_v^{(r)}) - c(T_w^{(r)}) \leq 2 + N_2 - 2(N_2 - k + g + 1) = -N_2 + 2k - 2g \leq D$. Otherwise $T[U]$ is a path with r as one of the endvertices, say $rv_1v_2 \dots v_t$ for some integer $t \geq 0$. If $t > D$, then $c(T_{v_1}^{(r)}) \geq \sum_{i=1}^{t-1} c(v_i) + c(T_{v_t}^{(r)}) \geq D + c(T_{v_t}^{(r)}) \geq D + (N_2 - k + g + 1) \geq k - g + 1$ which contradicts the definition of the support subtree T^* as in this case v_1 should be in $V(T^*)$. If $t = D$, similarly as above, we have $c(T_{v_1}^{(r)}) \geq k - g$

and hence, by Lemma 1.3, r is covered by some discharge from \mathcal{Q} . In any case, we have that there are at most D vertices which are not covered by any discharge from \mathcal{Q} , i.e. $|\bigcup_{(S,l,M,n) \in \mathcal{Q}} (M \cup \{n\})| \geq |V(T)| - D$.

If $|V(T^*)| = 1$ and $N_2 \geq 2k - 2g$, let r be the only support vertex and $r \neq v \in V(T)$ be a vertex in T . By the definition of the support subtree, we have that $c(T_v^{(r)}) \leq k - g$. Let w be the parent of v in $T^{(r)}$. We have $c(T[vw; w]) = N_2 - c(T_v^{(r)}) \geq (2k - 2g) - (k - g) = k - g$. Therefore $V(T) - r \subseteq \bigcup_{(S,l,M,n) \in \mathcal{Q}} (M \cup \{n\})$. \square

1.5. Proof of the Lemma 1.1

In this section we prove Lemma 1.1. We first consider the case that $N_2 \geq 2k - 2g$. If $|V(T^*)| = 1$, let r be the only support vertex. If $c(r) < g + 1$, then by Lemmas 1.7 and 1.5 and the condition that $2g + h > 3$, we have $|V_1| \leq \sum_{(S,l,M,n) \in \mathcal{Q}} |M \cap V_1| + 1 \leq \sum_{(S,l,M,n) \in \mathcal{Q}} (c(l) - g - 1) + 1 \leq \sum_{i \geq g+1} (i - g - 1) |V_i| + 1 < \sum_{i \geq g+1} (i - g - 1) |V_i| + 2g + h - 2$. Otherwise, $c(r) \geq g + 1$ and r can be an overloading vertex and we apply Lemmas 1.6 (take $T_0 := T$) to bound the corresponding discharges as follows: $|V_1| \leq \sum_{(S,l,M,n) \in \mathcal{Q}, l \neq r} |M \cap V_1| + \sum_{(S,l,M,n) \in \mathcal{Q}, l=r} |M \cap V_1| \leq \sum_{(S,l,M,n) \in \mathcal{Q}, l \neq r} (c(l) - g - 1) + \max\{0, c(r) - g - 1, c(r) + N_2 - 2k - 2\} \leq \sum_{(S,l,M,n) \in \mathcal{Q}, l \neq r} (c(l) - g - 1) + c(r) + g + h - 4 \leq \sum_{i \geq g+1} (i - g - 1) |V_i| + 2g + h - 3$. The third inequality follows from the condition that $N_2 \leq 2k + g + h - 2$. In any case the inequality (\diamond) is contradicted.

If $|V(T^*)| > 1$, then, by Lemma 1.7, all vertices in V_1 are covered by some discharge from \mathcal{Q} . For a vertex $u \in V(T^*)$, by Lemma 1.6 (take $T_0 := T^*[u]$) and Observation 1.2, we have

$$\sum_{(S,u,M,n) \in \mathcal{Q}} |M \cap V_1| \leq \max\{0, c(T^*[u]) - k - 1\} + d_{T^*}(u) \max\{0, c(u) - g - 1\}.$$

Define U_1 to be the set of vertices $u \in V(T^*)$ satisfying $d_{T^*}(u) = 1$, U_2 the set of vertices $u \in V(T^*)$ satisfying $d_{T^*}(u) > 1$ and $c(T^*[u]) \geq k + 1$, and U_3 the set of vertices $u \in V(T^*)$ satisfying $c(u) \geq g + 1$. Recall that U_1 is exactly the set of support vertices and $c(T^*[u]) \geq k + 1$ for all $u \in U_1$. As U_1 is disjoint with U_2 , we have $N_2 \geq \sum_{u \in U_1 \cup U_2} c(T^*[u]) + \sum_{u \in U_3 - (U_1 \cup U_2)} c(u)$, and

$$\begin{aligned} & \sum_{(S,u,M,n) \in \mathcal{Q}, u \in V(T^*)} |M \cap V_1| \\ \leq & \sum_{u \in U_1 \cup U_2} (c(T^*[u]) - k - 1) + \sum_{u \in U_3} d_{T^*}(u) (c(u) - g - 1) \\ = & \sum_{u \in U_1 \cup U_2} (c(T^*[u]) - k - 1) + \sum_{u \in U_3} (c(u) - g - 1) + \sum_{u \in U_3 - U_1} (d_{T^*}(u) - 1) (c(u) - g - 1) \\ \leq & \sum_{u \in U_1 \cup U_2} c(T^*[u]) + \sum_{u \in U_3 - (U_1 \cup U_2)} c(u) + \sum_{u \in U_3} (c(u) - g - 1) + \sum_{u \in U_1} (-k - 1) \\ & + \sum_{u \in U_3 \cap U_2} ((d_{T^*}(u) - 1) (c(u) - g - 1) - k - 1) + \sum_{u \in U_3 - (U_1 \cup U_2)} ((d_{T^*}(u) - 1) (c(u) - g - 1) - c(u)) \\ \leq & N_2 + \sum_{u \in U_3} (c(u) - g - 1) + \sum_{u \in U_3 - U_1} (d_{T^*}(u) - 2) (-k - 1) + 2(-k - 1) \\ & + \sum_{u \in U_3 - U_1} (d_{T^*}(u) - 2) (c(u) - g - 1) \\ \leq & \sum_{u \in U_3} (c(u) - g - 1) + N_2 - 2k - 2. \end{aligned}$$

In the third inequality we utilize the basic fact about tree that $\sum_{u \in U_1} 1 = \sum_{u \in U_1} d_{T^*}(u) = \sum_{u \in V(T^*) - U_1} (d_{T^*}(u) - 2) + 2$. Thus we have

$$\begin{aligned}
|V_1| &\leq \sum_{(S,l,M,n) \in \mathcal{Q}, l \notin V(T^*)} |M \cap V_1| + \sum_{(S,l,M,n) \in \mathcal{Q}, l \in V(T^*)} |M \cap V_1| \\
&\leq \sum_{(S,l,M,n) \in \mathcal{Q}, l \notin V(T^*)} (c(l) - g - 1) + \sum_{l \in U_3} (c(l) - g - 1) + N_2 - 2k - 2 \\
&\leq \sum_{i \geq g+1} (i - g - 1) |V_i| + g + h - 4,
\end{aligned}$$

which contradicts the inequality (\diamond) .

We now consider the case that $N_2 < 2k - 2g$. Note that in this case $|V(T^*)| = 1$ always holds. Let r be the only support vertex. Set $D := 2g + h - 3 > 0$ in Lemma 1.7, we have $|V_1| \leq \sum_{(S,l,M,n) \in \mathcal{Q}, l \neq r} (c(l) - g - 1) + \max\{0, c(r) - g - 1, c(r) + N_2 - 2k - 2\} + D \leq \sum_{(S,l,M,n) \in \mathcal{Q}, l \neq r} (c(l) - g - 1) + \max\{0, c(r) - g - 1, c(r) - 2g - 3\} + 2g + h - 3 \leq \sum_{i \geq g+1} (i - g - 1) |V_i| + 2g + h - 3$, which contradicts the inequality (\diamond) .

Thus it is proved the existence of a subtree S with weight $k - g + 1 \leq c(S) \leq k$. As Assumption (Ω) cannot hold, it is not hard to see that the subtree S can be found by the iterative overload-discharge process described in Algorithm 1. The cycle C_T and the mapping ρ can be constructed in $O(N_1)$ time [Hie73]. Note that there are $O(|V(C_T)|)$, i.e. $O(N_1)$ iterations, since the initial vertex $v \in V(C_T)$ can be revisited at most once. Thus S can be computed in $O(N_1)$ time. This completes the proof of Lemma 1.1.

We remark that here we assume that each arithmetic operation be done in constant time. If the arithmetic operations require logarithmic cost, then one can have $O(\Delta(T) \cdot N_1 \log \frac{N_2}{N_1})$ running time, where $\Delta(T)$ denotes the maximum degree of T .

Algorithm 1

Input: A tree T of N_1 vertices and vertex weights $c : V(T) \rightarrow \mathbb{N}$ with $c(T) = N_2$ such that $1 \leq k \leq N_2$, $g + h > 2$ and $2k - 4g - h + 3 \leq N_2 \leq 2k + g + h - 2$ ($k, g, N_1, N_2 \in \mathbb{N}$), where $h := 2N_1 - N_2$, and $c(v) \leq k$ for any $v \in V(T)$.

Output: A subtree S of T with $k - g + 1 \leq c(S) \leq k$.

- 1 Construct the directed cycle C_T and the homomorphism $\rho : V(C_T) \rightarrow V(T)$. Choose an arbitrary vertex v of C_T . Set $s := v$ and $t := v$.
 - 2 **while** $c([s, t]) < k - g + 1$ **do**
 - 3 \lfloor Set $t := t^+$.
 - 4 **if** $c([s, t]) \leq k$ **then**
 - 5 \lfloor Output $\rho([s, t])$.
 - 6 **while** $c([s, t]) > k$ **do**
 - 7 \lfloor Set $s := s^+$.
 - 8 **if** $c([s, t]) \geq k - g + 1$ **then**
 - 9 \lfloor Output $\rho([s, t])$.
 - 10 **go to** 2.
-

1.6. Some Examples

In this section we give some examples and show that the conditions in Lemma 1.1 are tight from several aspects. A useful fact to study some examples mentioned below is that Algorithm 1 is an exhaustive search when the input tree is a path.

The condition $1 \leq k \leq N_2$ should clearly be included for our interest.

We show that the condition $2g + h > 3$ is tight. Consider the *star* T of order $2p$ for some $p > 1$, such that center vertex has weight 1 and the other $2p - 1$ vertices have weight 2. We have $N_2 = 2N_1 - 1 = 4p - 1$ and $h = 1$. Set $k := N_1 = 2p \geq 4$ and $g := 1$. One can easily check that all conditions are satisfied except that $2g + h = 3$, and T has no subtree of weight k .

For the condition $N_2 \geq 2k - 4g - h + 3$, we consider the path $v_1v_2 \dots v_{p+2q}$ of order $p + 2q$ for some integers $p > 1$ and $q \geq 1$. Set $c(v_{q+i}) := 1$ for $i = 1, 2, \dots, p$, and $c(v_j) := 2$ for any $j \neq q + 1, q + 2, \dots, q + p$. We have $N_2 = p + 4q$ and $h = p$. Set $k := p + 2q + 1$ and $g := 1$. One can easily check that all conditions are satisfied except that $N_2 = 2k - 4g - h + 2$, and T has no subtree of weight k .

We next discuss the condition $N_2 \leq 2k + g + h - 2$. Let $p > 1$ be an integer, and T be a path $v_1v_2 \dots v_{2p+3}$ of order $2p + 3$. Set $c(v_{p+2}) := p + 2$, $c(v_{p+i}) := 2$ for $i = 1, 3$, and $c(v_j) := 1$ for any $j \neq p + 1, p + 2, p + 3$. We have $N_2 = 3p + 6$ and $h = 2N_1 - N_2 = p$. Set $k := p + 3$ and $g := 1$. One can easily check that all conditions are satisfied except that $N_2 = 2k + g + h - 1$, and T has no subtree of weight k .

As we have seen, the condition $c(v) \leq k$ for all $v \in V(T)$ is one of the key ingredients to make the overload-discharge process work. If this condition is violated, then the existence of a subtree of the desired weight cannot be assured. Let T be the star of order $p + 1$ for some integer $p > 1$. We set the vertex weight of the center vertex to be $q + 1$ for some integer $2 < q < p + 2$, and those of the other p vertices (leaves) to be 1. We have $N_2 = p + q + 1$ and $h = p - q + 1$. Set $k := q$ and $g := 2$. Then it is clear that all conditions are satisfied except that there exists a vertex (the center vertex) with weight larger than k , and T has no subtree of weight between $k - g + 1$ and k .

Cycles of Specified Length

2.1. Overview of Cycle Spectra of Planar Graphs

The *cycle spectrum* $CS(G)$ of a simple graph G is defined to be the set of integers k for which there is a cycle of length k in G . G is said to be *hamiltonian* if $|V(G)| \in CS(G)$ and *pancyclic* if its cycle spectrum has all possible lengths, i.e. $CS(G) = \{3, \dots, |V(G)|\}$. Cycle spectra of graphs have been extensively studied in many directions, in this paper we study cycle spectra of planar hamiltonian graphs with minimum degree $\delta \geq 4$.

In 1956, Tutte [Tut56] proved his seminal result that every 4-connected planar graph is hamiltonian. Motivated by Tutte's theorem together with the metaconjecture proposed by Bondy [Bon75] that almost any non-trivial conditions for hamiltonicity of a graph should also imply pancyclicity, Bondy [Bon75] conjectured in 1973 that every 4-connected planar graph G is *almost pancyclic*, i.e. $|CS(G)| \geq |V(G)| - 3$, and Malkevitch [Mal88] conjectured in 1988 that every 4-connected planar graph is pancyclic if it contains a cycle of length 4 (see [Mal71, Mal78] for other variants).

These two conjectures remain open, while 4 is the only known cycle length that can be missing in a cycle spectrum of a 4-connected planar graph. For example, the line graph of a cyclically 4-edge-connected cubic planar graph with girth at least 5 is a 4-regular 4-connected planar graph with no cycle of length 4, see also [Mal71, Tre89]. If we relax the connectedness, more cycle lengths are known for being absent in some cycle spectra. Choudum [Cho77] showed that for every integer $k \geq 7$, there exist 4-regular 3-connected planar hamiltonian graphs of order larger than k each has cycles of all possible lengths except k , which means every integer $k \geq 7$ can be absent in the cycle spectra of some 4-regular 3-connected planar hamiltonian graphs. Another interesting example was constructed by Malkevitch [Mal71], which is, for every $p \in \mathbb{N}$, a 4-regular planar hamiltonian graph G of order $|V(G)| = 6p$ whose cycle spectrum $CS(G) = \{3, 4, 5, 6\} \cup \{r \in \mathbb{N} : \frac{|V(G)|}{2} \leq r \leq |V(G)|\}$, as shown in Figure 2.1.

So far we have seen which cycle lengths can be absent in some cycle spectra, we now ask the opposite question, i.e. which cycle lengths must be present in all cycle spectra. It is known that every planar graph with $\delta \geq 4$ must contain cycles of length 3, 5 [WL02] and 6 [FJMv02], which is shown to be best possible by the aforementioned examples. It is also known that every 2-connected planar graph with $\delta \geq 4$ must have a cycle of length 4 or 7 [HK08], a cycle of length 4 or 8 and a cycle of length 4 or 9 [MT]. While the presence of a cycle of length 3 follows easily from Euler's formula, the rest of them were shown by the discharging method.

Another powerful tool in searching cycles of specified length is the so-called Tutte path method, which was first introduced by Tutte in his proof of hamiltonicity of 4-connected planar graphs. Using this technique, Nelson (see [Plu75, Tho83]), Thomas and Yu [TY94] and Sanders [San97] showed that every 4-connected planar graph contains cycles of length $|V(G)| - 1$, $|V(G)| - 2$ and $|V(G)| - 3$, respectively. Note that we always assume $k \geq 3$

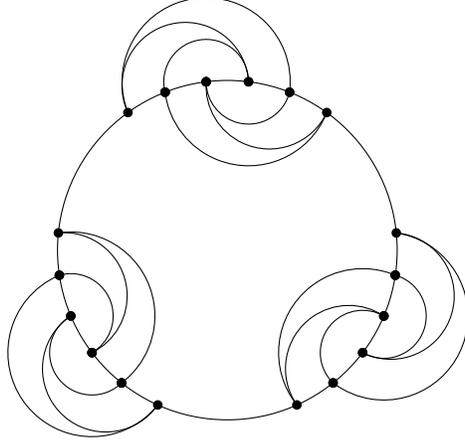


Figure 2.1. A 4-regular planar hamiltonian graph G which has no cycle of length between 7 and $\frac{|V(G)|}{2} - 1$.

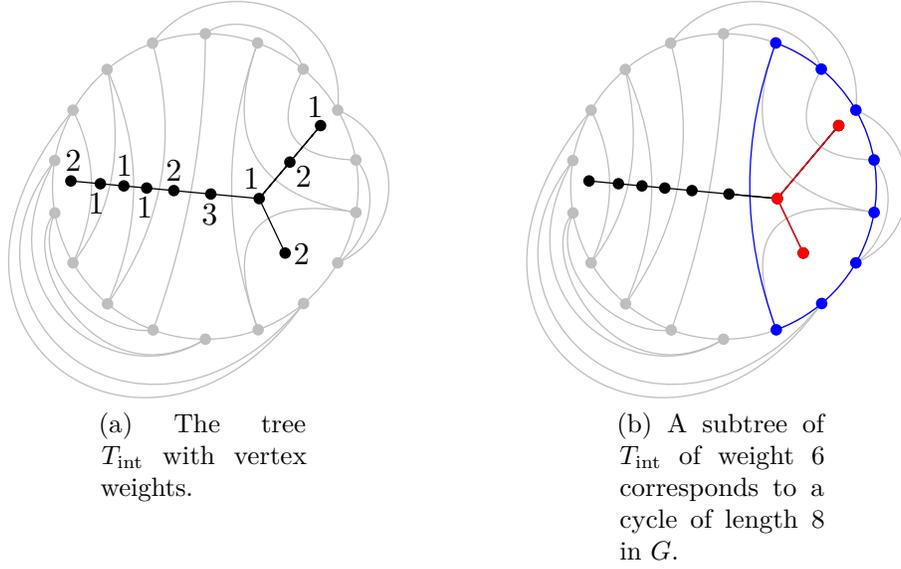
when we say a graph contains a cycle of length k . Chen et al. [CFY04] noticed that the Tutte path method cannot be generalized for smaller cycle lengths, they hence combined Tutte paths with contractible edges and showed the existence of cycles of length $|V(G)| - 4$, $|V(G)| - 5$ and $|V(G)| - 6$. Following this approach, Cui et al. [CHW09] showed that every 4-connected planar graph has a cycle of length $|V(G)| - 7$. To summarize, every 4-connected planar graph contains a cycle of length k for every $k \in \{|V(G)|, |V(G)| - 1, \dots, |V(G)| - 7\}$ with $k \geq 3$.

2.2. Cycles of Length Close to Medium-Length

With the knowledge of these short and long cycles, Mohr [Moh18] asked whether cycles of length close to $\frac{|V(G)|}{2}$ also exist, and he answered his question by showing that every planar hamiltonian graph G satisfying $|E(G)| \geq 2|V(G)|$ has a cycle of length between $\frac{1}{3}|V(G)|$ and $\frac{2}{3}|V(G)|$. We present his simple and elegant argument in the following.

Let G^* be the dual graph of the plane graph G and C be a *Hamilton cycle* (i.e. a spanning cycle) of G . Note that C separates the Euclidean plane into two open regions C_{int} and C_{ext} containing no vertex. Let G_{int} and G_{ext} be the graphs obtained from G by deleting the edges in C_{ext} and in C_{int} , respectively. We always assume that $|E(G_{\text{int}})| \geq |E(G_{\text{ext}})|$. As C is a Hamilton cycle, its dual disconnects G^* into two trees say T_{int} lying on C_{int} and T_{ext} on C_{ext} (see Figure 2.2). By Euler's formula, we have $|V(G^*)| = |E(G)| - |V(G)| + 2$, and hence $|V(T_{\text{int}})| \geq \frac{1}{2}|V(G^*)| \geq \frac{1}{2}|V(G)| + 1$. We define vertex weight $c(v) := d_{G^*}(v) - 2 \geq 1$ for every vertex $v \in V(G^*) \supset V(T_{\text{int}})$, where $d_{G^*}(v)$ is the degree of v in G^* , or equivalently, the face length of v in G (see Figure 2.2(a)). It is not hard to see that for every subtree S of T_{int} , the set of edges of G^* having exactly one endvertex in S is indeed the dual of an edge set of a cycle in G of length $c(S) + 2$, where $c(S) := \sum_{v \in V(S)} c(v)$ (see Figure 2.2(b)).

Thus the problem of finding a cycle of specified length is transformed to the problem of finding a subtree of specified weight: the existence of a subtree S of weight k in T_{int} implies the existence of a cycle of length $k + 2$ in G . It is left to show that there is a subtree S in T_{int} with $\frac{1}{3}|V(G)| - 2 \leq c(S) \leq \frac{2}{3}|V(G)| - 2$. First note that $c(v) \leq \frac{1}{2}|V(G)| - 2$ for all $v \in V(T_{\text{int}})$; otherwise $c(T_{\text{int}}) > \frac{1}{2}|V(G)| - 2 + |V(T_{\text{int}})| - 1 \geq |V(G)| - 2$, which is not possible as T_{int} corresponds to the Hamilton cycle of length $|V(G)|$. If there is a vertex

Figure 2.2. The tree T_{int} in the dual graph of G .

$v \in V(T_{\text{int}})$ with $c(v) \geq \frac{1}{3}|V(G)| - 2$, then we can simply take S to be this single vertex v . Suppose $c(v) < \frac{1}{3}|V(G)| - 2$ for all $v \in V(T_{\text{int}})$. We take S to be a maximal subtree of T_{int} with $c(S) \leq \frac{2}{3}|V(G)| - 2$, it is clear that $c(S) \geq \frac{1}{3}|V(G)| - 2$. Thus G has a cycle of length between $\frac{1}{3}|V(G)|$ and $\frac{2}{3}|V(G)|$.

2.3. Planar Hamiltonian Graphs

We recapitulate the main content of Mohr's inspiring proof. Given a planar hamiltonian graph G , we can have a tree T (in the dual graph) of at least $\frac{1}{2}|E(G)| - \frac{1}{2}|V(G)| + 1$ vertices with vertex weights $c : V(T) \rightarrow \mathbb{N}$ such that $c(T) = |V(G)| - 2$ and $c(v) \leq c(T) - |V(T)| + 1 \leq \frac{3}{2}|V(G)| - \frac{1}{2}|E(G)| - 2$ for all $v \in V(T)$. And, if there is a subtree of weight k in T , then there is a cycle of length $k + 2$ in G .

Combining Mohr's transformation and Lemma 1.1 we have the following:

COROLLARY 2.1. *Let G be a planar hamiltonian graph with $|E(G)| \geq (2 + \gamma)|V(G)|$ for some real number $-1 \leq \gamma < 1$. Let $k, g \in \mathbb{N}$ such that $g + \lceil \gamma|V(G)| \rceil + 2 > 0$, $3 \leq k \leq |V(G)|$ and $\lfloor \frac{(1-\gamma)|V(G)|}{2} \rfloor \leq k \leq \frac{\lceil (1+\gamma)|V(G)| \rceil}{2} + 2g + \frac{3}{2}$. There exists a cycle K in G of length $k - g + 1 \leq |V(K)| \leq k$, and K can be found in linear time if a Hamilton cycle of G is given.*

PROOF. Let T be the tree with vertex weights c that we mentioned before. We set $\tilde{k} := k - 2 \geq 1$ and $h := \lceil \gamma|V(G)| \rceil + 4$. We check the conditions required for applying Lemma 1.1 on the parameters \tilde{k}, g, h, N_1 and N_2 as follows. First we have that $g + h > 2$, $1 \leq \tilde{k} \leq N_2 = |V(G)| - 2$, $2\tilde{k} \leq \lceil (1 + \gamma)|V(G)| \rceil + 4g - 1 = N_2 + 4g + h - 3$, and $\tilde{k} \geq \lfloor \frac{(1-\gamma)|V(G)|}{2} \rfloor - 2 \geq \frac{\lceil (1-\gamma)|V(G)| \rceil}{2} - \frac{1}{2} - 2 = \frac{|V(G)| - \lceil \gamma|V(G)| \rceil}{2} - \frac{1}{2} - 2 \geq \frac{N_2}{2} - \frac{g}{2} - \frac{h}{2} + 1$. Note also that $2|V(T)| \geq |E(G)| - |V(G)| + 2 \geq (1 + \gamma)|V(G)| + 2 = c(T) + \gamma|V(G)| + 4$ implies $2N_1 \geq N_2 + h$, and for every $v \in V(T)$, $c(v) \leq \frac{3}{2}|V(G)| - \frac{1}{2}|E(G)| - 2 \leq \frac{3}{2}|V(G)| - \frac{2+\gamma}{2}|V(G)| - 2 = \frac{1-\gamma}{2}|V(G)| - 2$ implies $c(v) \leq \lfloor \frac{(1-\gamma)|V(G)|}{2} \rfloor - 2 \leq \tilde{k}$. As all conditions are satisfied, by Lemma 1.1, there exists a subtree S of T of weight $\tilde{k} - g + 1 \leq c(S) \leq \tilde{k}$ which

can be found in linear time. And hence G has a cycle K of length $k - g + 1 \leq |V(K)| \leq k$ which can be found in linear time provided a Hamilton cycle of G is given, since every planar graph can be embedded in plane in linear time [CNAO85] and the tree T_{int} can then be easily constructed from the planar embedding in linear time. \square

We specify some implications as follows.

COROLLARY 2.2. *Every planar hamiltonian graph G with $\delta(G) \geq 3$ has a cycle of length $\lfloor \frac{|V(G)|+1}{4} \rfloor + 2 \leq k \leq \lfloor \frac{3|V(G)|}{4} \rfloor$. Every planar hamiltonian graph G with $\delta(G) \geq 4$ has a cycle of length k for every $k \in \{\lfloor \frac{|V(G)|}{2} \rfloor, \dots, \lceil \frac{|V(G)|}{2} \rceil + 3\}$ with $3 \leq k \leq |V(G)|$. Every planar hamiltonian graph G with $\delta(G) \geq 5$ has a cycle of length k for every $k \in \{\lfloor \frac{|V(G)|}{4} \rfloor, \dots, \lceil \frac{3|V(G)|}{4} \rceil + 3\}$ with $3 \leq k \leq |V(G)|$. Each of these cycles can be found in linear time if a Hamilton cycle of G is given.*

PROOF. It follows immediately when we in Theorem 2.1 set $\gamma := -\frac{1}{2}$, $g := \lfloor \frac{|V(G)|}{2} \rfloor - 1$ and $k := \lfloor \frac{3|V(G)|}{4} \rfloor$; $\gamma := 0$ and $g := 1$; and $\gamma := \frac{1}{2}$ and $g := 1$, respectively. \square

It is known that a Hamilton cycle can be found in linear time for every 4-connected planar graph [CN89]. Thus those cycles mentioned above can be simply found in linear time each in this case.

2.4. 3-Connected Planar Hamiltonian Graphs

Note that Malkevitch's example (see Figure 2.1) illustrates that not every planar hamiltonian graph G with $\delta \geq 4$ can have a cycle of length $\lfloor \frac{|V(G)|}{2} \rfloor - 1$ or $\lfloor \frac{|V(G)|}{2} \rfloor - 2$. As a further application we prove in this section that this cycle length can be assured for 3-connected planar hamiltonian graphs with $\delta \geq 4$.

THEOREM 2.3. *Let G be a 3-connected planar hamiltonian graph with minimum degree $\delta(G) \geq 4$. If $|V(G)| \geq 8$ is even, there exists a cycle of length either $\frac{1}{2}|V(G)| - 2$ or $\frac{1}{2}|V(G)| - 1$ in G , and it can be found in linear time if a Hamilton cycle is given.*

PROOF. We adopt the notations defined in Section 2.2. If every face of G_{int} is of length either $|V(G)|$ or less than $\frac{1}{2}|V(G)|$, i.e. $c(v) \leq \frac{1}{2}|V(G)| - 3$ for every $v \in V(T_{\text{int}})$, by Lemma 1.1 (set $g := 2$ and $h := 4$), there exists a subtree of weight either $\frac{1}{2}|V(G)| - 4$ or $\frac{1}{2}|V(G)| - 3$ in T_{int} and hence a cycle of length either $\frac{1}{2}|V(G)| - 2$ or $\frac{1}{2}|V(G)| - 1$ in G .

Recall that $|E(G_{\text{int}})| \geq \frac{3}{2}|V(G)|$. If $|E(G_{\text{int}})| > \frac{3}{2}|V(G)|$, then $|V(T_{\text{int}})| \geq \frac{1}{2}|V(G)| + 2 = \frac{1}{2}c(T_{\text{int}}) + 3$ and $c(v) \leq c(T_{\text{int}}) - |V(T_{\text{int}})| + 1 \leq \frac{1}{2}|V(G)| - 3$ for all $v \in V(T_{\text{int}})$. By Lemma 1.1 (set $g := 1$ and $h := 6$), there exists a subtree of weight $\frac{1}{2}|V(G)| - 3$ in T_{int} and hence a cycle of length $\frac{1}{2}|V(G)| - 1$ in G .

Now we can assume that $|E(G_{\text{int}})| = \frac{3}{2}|V(G)|$ and G_{int} has a face of length $\frac{1}{2}|V(G)|$. It holds immediately that $|E(G_{\text{ext}})| = \frac{3}{2}|V(G)|$ since $|E(G_{\text{int}})| + |E(G_{\text{ext}})| = |E(G)| + |V(G)| \geq 3|V(G)|$ and $|E(G_{\text{int}})| \geq |E(G_{\text{ext}})|$. And we can also assume that G_{ext} has a face of length $\frac{1}{2}|V(G)|$. In this case we have $d_G(v) = 4$ and $d_{G_{\text{int}}}(v) + d_{G_{\text{ext}}}(v) = 6$ for every $v \in V(G)$, and that there are exactly one face of length $|V(G)|$, one face of length $\frac{1}{2}|V(G)|$ and $\frac{1}{2}|V(G)|$ faces of length 3 in each of G_{int} and G_{ext} . We denote by F_{int} and F_{ext} be the faces of length $\frac{1}{2}|V(G)|$ in G_{int} and G_{ext} , respectively.

We claim that G is the square of a cycle of length $|V(G)|$, which is obtained from a cycle of length $|V(G)|$ by adding edges for every pair of vertices having distance 2 (see Figure 2.3).

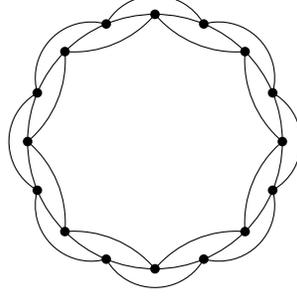


Figure 2.3. The square of a cycle of length 16.

It is obvious that the square of a cycle is pancyclic. We call a face of length 3 an i -triangle ($i = 0, 1, 2$) if it contains exactly i edges of the Hamilton cycle C . We assume that the plane graph G has the maximum number of 2-triangles over all of its planar embeddings. Let the Hamilton cycle C of G be $v_0v_1v_2 \dots v_{|V(G)|-1}v_0$ (indices modulo $|V(G)|$).

Suppose there is a 0-triangle $v_0v_iv_jv_0$ in G , say it is also in G_{int} , for some $0 < i-1 < j-2 < |V(G)| - 3$. If $i > 2$, then the face in G_{int} containing the path $v_1v_0v_iv_{i-1}$ is of length larger than 3 and smaller than $|V(G)|$. As there is exactly one such face in G_{int} , namely F_{int} , we can assume that $i = 2$ and $j = 4$. Then $d_{G_{\text{int}}}(v_1) = d_{G_{\text{int}}}(v_3) = 2$ and $d_{G_{\text{ext}}}(v_1) = d_{G_{\text{ext}}}(v_3) = 4$. Let v_{i_1}, v_{i_2} be the neighbors of v_1 other than v_0, v_2 , and v_{i_3}, v_{i_4} be the neighbors of v_3 other than v_2, v_4 , for some $2 < i_1 < i_2 < |V(G)|$ and $4 < i_3 < i_4 < |V(G)| + 2$.

If v_1 is adjacent to v_3 in G_{ext} , i.e. $i_1 = 3$ and $i_4 = |V(G)| + 1$, and the face in G_{ext} containing $v_{i_2}v_1v_3v_{i_3}$ is a face of length 3, i.e. $i_2 = i_3$, then it must be a 0-triangle. We can assume that $i_2 = i_3 = 5$. Clearly, $\{v_0, v_5\}$ is a separator of G , which contradicts that G is 3-connected. If v_1 is adjacent to v_3 in G_{ext} , but the face in G_{ext} containing $v_{i_2}v_1v_3v_{i_3}$ is a face of length larger than 3, then the faces in G_{ext} containing $v_0v_1v_{i_2}$ and $v_{i_3}v_3v_4$ must be 2-triangles and $\{v_{i_2}, v_{i_3}\} = \{v_{-1}, v_5\}$ is a separator of G , contradiction.

If v_1 is not adjacent to v_3 , then the face in G_{ext} containing $v_{i_1}v_1v_2v_3v_{i_4}$ is of length larger than 3, and hence $v_{-1}v_0v_1v_{-1}$ and $v_3v_4v_5v_3$ must be 2-triangles in G_{ext} . In this case we can swap v_0 and v_1 and swap v_3 and v_4 to obtain a planar embedding with more 2-triangles (see Figure 2.4(a)), which contradicts the maximality of the number of 2-triangles. Thus there is no 0-triangle in the plane graph G .

Suppose there is a 1-triangle $v_0v_1v_iv_0$ in G , say also in G_{int} , for some $2 < i < |V(G)| - 1$. It is not hard to see that we can assume that the face in G_{int} containing $v_0v_iv_{i+1}$ is F_{int} . Under this assumption we must have a sequence of $i - 1$ faces of length 3 such that all faces are 1-triangles except the last one which is a 2-triangle, namely $v_0v_1v_iv_0, v_1v_{i-1}v_iv_1, v_1v_2v_{i-1}v_1, \dots, v_{\lceil \frac{i}{2} \rceil - 1}v_{\lceil \frac{i}{2} \rceil}v_{\lceil \frac{i}{2} \rceil + 1}v_{\lceil \frac{i}{2} \rceil - 1}$.

We claim that $i \leq 4$. Suppose $i > 4$, we prove the claim for odd i , it can be proved for even i in a similar way. It is clear that $d_{G_{\text{ext}}}(v_{\lceil \frac{i}{2} \rceil - 3}) \leq 3$, $d_{G_{\text{ext}}}(v_{\lceil \frac{i}{2} \rceil - 1}) = 3$ and $d_{G_{\text{ext}}}(v_{\lceil \frac{i}{2} \rceil}) = 4$. Let v_{i_1}, v_{i_2} be the neighbors of $v_{\lceil \frac{i}{2} \rceil}$ other than $v_{\lceil \frac{i}{2} \rceil - 1}, v_{\lceil \frac{i}{2} \rceil + 1}$, and v_{i_3} be the neighbor of $v_{\lceil \frac{i}{2} \rceil - 1}$ other than $v_{\lceil \frac{i}{2} \rceil - 2}, v_{\lceil \frac{i}{2} \rceil}, v_{\lceil \frac{i}{2} \rceil + 1}$, for some $i < i_1 < i_2 \leq i_3 \leq |V(G)|$. Note that the face in G_{ext} containing $v_{i_1}v_{\lceil \frac{i}{2} \rceil}v_{\lceil \frac{i}{2} \rceil + 1}v_{\lceil \frac{i}{2} \rceil + 2}$ is of length larger than 3. Therefore the face in G_{ext} containing $v_{i_3}v_{\lceil \frac{i}{2} \rceil - 1}v_{\lceil \frac{i}{2} \rceil}v_{i_2}$ and that containing $v_{\lceil \frac{i}{2} \rceil - 3}v_{\lceil \frac{i}{2} \rceil - 2}v_{\lceil \frac{i}{2} \rceil - 1}v_{i_3}$ must be of length 3. It implies that $v_{\lceil \frac{i}{2} \rceil - 3} = v_{i_3} = v_{i_2}$ and $d_{G_{\text{ext}}}(v_{\lceil \frac{i}{2} \rceil - 3}) \geq 4$, contradiction.

Now we consider the case when $i = 4$. It is clear that $d_{G_{\text{ext}}}(v_2) = 4$ and $d_{G_{\text{ext}}}(v_3) = 3$. Let v_{i_1} be the neighbor of v_3 other than v_1, v_2, v_4 , and v_{i_2}, v_{i_3} be the neighbors of v_2 other than v_1, v_3 , for some $4 < i_1 \leq i_2 < i_3 \leq |V(G)|$. If the face in G_{ext} containing $v_{i_1}v_3v_4$ is of length larger than 3, then $i_1 = i_2 = |V(G)| - 1$, $i_3 = |V(G)|$ and $\{v_{-1}, v_4\}$ is separator of G . If the face in G_{ext} containing $v_{i_1}v_3v_4$ is of length 3 but that containing $v_{i_1}v_3v_2v_{i_2}$ is of length larger than 3, then $i_1 = 5$, $i_2 = |V(G)| - 1$, $i_3 = |V(G)|$ and $\{v_{-1}, v_5\}$ is a separator of G . If the faces in G_{ext} containing $v_{i_1}v_3v_4$ and $v_{i_1}v_3v_2v_{i_2}$ are of length 3 but that containing $v_{i_2}v_2v_{i_3}$ is of length larger than 3, then $i_1 = i_2 = 5$, $i_3 = |V(G)|$ and $\{v_0, v_5\}$ is a separator of G . If the faces in G_{ext} containing $v_{i_1}v_3v_4$, $v_{i_1}v_3v_2v_{i_2}$ and $v_{i_2}v_2v_{i_3}$ are of length 3, then $i_1 = i_2 = 5$, $i_3 = 6$ and $\{v_0, v_6\}$ is a separator of G . In any case it contradicts that G is 3-connected.

Finally, we consider the case when $i = 3$. It is clear that $d_{G_{\text{ext}}}(v_1) = 3$ and $d_{G_{\text{ext}}}(v_2) = 4$. Let v_{i_1}, v_{i_2} be the neighbors of v_2 other than v_1, v_3 , and v_{i_3} be the neighbor of v_1 other than v_0, v_2, v_3 , for some $3 < i_1 < i_2 \leq i_3 < |V(G)|$. If the face in G_{ext} containing $v_{i_1}v_2v_3$ is of length larger than 3, then $i_1 = |V(G)| - 2$ and $i_2 = i_3 = |V(G)| - 1$, which has been shown to be not possible. If the face in G_{ext} containing $v_{i_1}v_2v_3$ is of length 3 but that containing $v_{i_1}v_2v_{i_2}$ is of length larger than 3, then $i_1 = 4$, $i_2 = i_3 = |V(G)| - 1$ and $d_{G_{\text{int}}}(v_0) = 4$. Let v_{i_4} be the neighbor of v_0 other than v_{-1}, v_1, v_3 for some $3 < i_4 \leq |V(G)| - 2$. If the faces in G_{int} containing $v_{-1}v_0v_{i_4}$ is of length larger than 3, then $i_4 = 4$, which has been shown to be not possible. Hence $v_{-1}v_0v_{i_4}v_{-1}$ is 2-triangle, $i_4 = |V(G)| - 2$ and $\{v_{-2}, v_4\}$ is a separator of G , which is not possible. If the faces in G_{ext} containing $v_{i_1}v_2v_3$ and $v_{i_1}v_2v_{i_2}$ are of length 3, then $i_1 = 4$ and $i_2 = 5$. Swapping v_2 and v_3 yields a planar embedding of more 2-triangles (see Figure 2.4(b)), which contradicts the maximality of the number of 2-triangles. Hence we can conclude that there is no 1-triangle in the plane graph G .

It is clear that G is the square of a cycle of length $|V(G)|$ if it has a planar embedding with neither 0- nor 1-triangle. To find a cycle of the desired length, one can apply Algorithm 1 for T_{int} if $|E(G_{\text{int}})| > \frac{3}{2}|V(G)|$, or if there is no face of length $\frac{1}{2}|V(G)|$ in G_{int} and G_{ext} , otherwise, do swaps of some vertex pairs at most once for each face to obtain a planar embedding of the square of a cycle of length $|V(G)|$ with neither 0- nor 1-triangle, then a cycle of length $\frac{1}{2}|V(G)|$ can be easily found in such planar embedding in linear time. \square

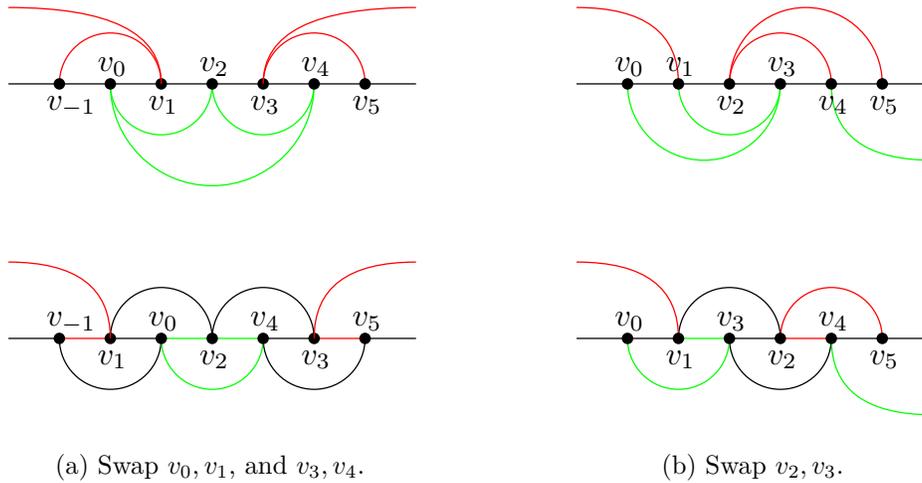


Figure 2.4. Swap vertices to obtain planar embedding of more 2-triangles.

Part II

CHAPTER 3

Generalized Cut Trees for Edge-Connectivity

3.1. Introduction

We propose a general notion of cut trees that *cover* some binary relation R on the vertices. We consider only binary relations R that are irreflexive and symmetric, which allows us to see R as a set of unordered pairs $\{a, b\}$ satisfying $a \neq b$. We will study cut trees (see Definition 3.1) for three relations, each of them giving structural insights about the edge-connectivity in graphs.

We call a pair $\{v, w\}$ of vertices *pendant* if $\lambda_G(v, w) = \min\{d_G(v), d_G(w)\}$. The study of pendant pairs is motivated by the well-known, simple and widely used min-cut algorithm of Nagamochi and Ibaraki [NI92], which refines the work of Mader [Mad73, Mad71] in the early 70s, and was simplified by Stoer and Wagner [SW97] and Frank [Fra94]. The key approach of this algorithm is to iteratively contract a pendant pair of the input graph $G := (V(G), E(G))$ in near-linear time by using *maximal adjacency orderings* (also known as *maximum cardinality search* [TY84]). Having done that $n - 2$ times, where $n := |V(G)|$, one can obtain a min-cut by just considering the minimum degree of all intermediate graphs.

As early as 1973, and originally motivated by the structure of minimally k -edge-connected graphs, Mader proved that every graph with minimum degree $\delta \geq 1$ contains a pendant pair [Mad73]. This holds also for the vertex-connectivity variant of pendant pairs, which nowadays is most easily proven by using maximal adjacency orderings. Later, Mader improved his result by showing that every simple graph with minimum degree δ contains in fact $\frac{\delta(\delta+1)}{2}$ pendant pairs [Mad74]. By considering the cut tree covering non-pendant pairs, it was recently improved by Schmidt and the author [LS18]. They showed that every simple graph that satisfies $\delta \geq 5$ or $\lambda \geq 4$ or $\kappa \geq 3$ contains $\frac{\delta n}{30}$ pendant pairs. This result is optimal up to a constant factor and that every of the three assumptions is best possible. They also showed how to compute these pendant pairs from a Gomory-Hu tree in linear time. We will present this result in Section 3.3 and give an improvement of the lower bound by a constant factor, namely we show that under the same conditions there are at least $\frac{\delta n}{24}$ pendant pairs.

In Section 3.4, we study two other cut trees, from which results on sparsification done by vertex subset contractions will be derived. We consider the cut tree covering the vertex pairs $\{v, w\}$ for which $\lambda_G(v, w) < \delta$ in Section 3.4.1. We prove that every simple graph G with $\delta > 0$ has $O(n/\delta)$ δ -edge-connected components, and contracting these components leaves $O(n)$ edges. For a simple graph G satisfying $0 < \lambda < \delta$, it was recently shown [LST18] that G has $O((n/\delta)^2)$ min-cuts. We strengthen this and show that G has $O((n/\delta)^2)$ cuts of size less than $\min\{\frac{3}{2}\lambda, \delta\}$, and $O((n/\delta)^{\lfloor 2\alpha \rfloor})$ cuts of size not larger than $\min\{\alpha\lambda, \delta - 1\}$ for any given α , respectively.

Recently, Kawarabayashi and Thorup [KT18] gave the first deterministic near-linear time algorithm for finding a min-cut of G . Subsequently, Henzinger, Rao and Wang [HRW17] obtained an improved variant with running time $O(m \log^2 n \log \log^2 n)$ by replacing the

diffusion subroutine with a flow-based one, where $m := |E(G)|$. A crucial step in both algorithms is a sparsification routine [KT18, Theorem 3] for large minimum degree δ that contracts vertex subsets of G such that, after these sparsifications, the remaining graph has only $O((n \log^c n)/\delta)$ vertices and $O(n \log^c n)$ edges (for some constant c) and all non-trivial min-cuts of G are preserved. Schmidt, Thorup and the author [LST18] later showed that, for a simple graph G satisfying $\delta > 0$, we can find some vertex subsets in near-linear time such that all non-trivial min-cuts are preserved, and only $O(n/\delta)$ vertices and $O(n)$ edges are left when these vertex subsets are contracted. This eliminates the poly-logarithmic factor needed above. Here, we introduce the cut tree that covers the vertex pairs that are separated by some non-trivial min-cut, and give an alternative proof of this new result (see Section 3.4.2). We also show that such a cut tree exists and can be computed in near-linear time for every simple graph G satisfying $\lambda \neq 0, 2$.

A Note on the History of Maximal Adjacency Orderings. Mader’s proof for the existence of one pendant pair relies strongly on [Mad71, Lemma 1], which in turn uses special orderings on the vertices. Interestingly, these orderings are *maximal adjacency orderings* and this fact exhibits a nowadays almost forgotten variant of them, which existed long before they got 1984 their first name (maximum cardinality search [TY84]). We are only aware of one place in literature where this is (briefly) mentioned: [Mad96, p. 443]. Mader’s existential proof can in fact be made algorithmic. A direct comparison between the old and the modern variant however shows that the modern maximal adjacency orderings are nicer to describe, as they work on the original graph, while Mader iteratively moves edges in the graph in order to represent the essential connectivity information on the already visited vertex set with a clique.

3.2. Cut Trees

We define our main tool in this chapter in the following. Let \mathcal{T} be a tree whose vertex set is a partition of $V(G)$. For the sake of notational clarity, we will call the vertices of such trees *nodes*. Let $AB \in E(\mathcal{T})$ and let C_{AB} be the union of the nodes that are contained in the component of $\mathcal{T} - AB$ containing A , and symmetrically, $C_{BA} = \overline{C_{AB}}$. For an edge $AB \in E(\mathcal{T})$, let $c(AB) := d_G(C_{AB})$ be the *size* of its corresponding edge-cut in G .

DEFINITION 3.1. Given an undirected graph $G := (V, E)$ and a binary relation R on $V(G)$, a *cut tree* \mathcal{T} *covering* R is a tree whose vertex set is a partition of $V(G)$, such that

- (i) for every $A \in V(\mathcal{T})$ and every $a, a' \in A$, we have $\{a, a'\} \notin R$,
- (ii) for every tree edge $AB \in E(\mathcal{T})$, there exist $a \in A$ and $b \in B$ that satisfy $\{a, b\} \in R$ and
- (iii) for every tree edge $AB \in E(\mathcal{T})$, there exist $a^* \in A$ and $b^* \in B$ that satisfy $\lambda_G(a^*, b^*) = d_G(C_{AB})$, i.e. C_{AB} is a minimum a^* - b^* -cut in G .

This definition generalizes the well-known Gomory-Hu trees, as a Gomory-Hu tree is a cut tree covering the maximal binary relation on $V(G)$ (i.e. $\{\{v, w\} : v, w \in V(G), v \neq w\}$); and also the tool *tree representations* used in [Cai93, Mad95]. By choosing the binary relation R in the above cut-tree appropriately, we will prove results about the edge-connectivity structure of graphs in the remaining sections.

3.3. Pendant Trees and Pendant Pairs

We call a cut tree \mathcal{T} covering the set of all pairs of non-pendant vertices a *pendant tree*. By Definition 3.1, we have that (i) every pair of two distinct vertices in a common node in $V(\mathcal{T})$ is pendant, (ii) for every edge $AB \in E(\mathcal{T})$, there are vertices $a \in A$ and $b \in B$ such that (a, b) is non-pendant, and (iii) for every edge $AB \in E(\mathcal{T})$, there are vertices $a^* \in A$ and $b^* \in B$ such that $c(AB) = \lambda_G(a^*, b^*)$.

The following lemma allows us to find a non-pendant pair from two adjacent nodes of a pendant tree efficiently.

LEMMA 3.2. *Let AB be an edge of a pendant tree \mathcal{T} and let a_{\max} and b_{\max} be vertices in A and B of maximum degrees, respectively. Then (a_{\max}, b_{\max}) is non-pendant.*

PROOF. By Property (ii) of Definition 3.1, there are vertices $a \in A$ and $b \in B$ such that $\lambda(a, b) < \min\{d(a), d(b)\}$. Since (a, a_{\max}) and (b, b_{\max}) are pendant, a minimum a - b -cut can neither separate a from a_{\max} nor b from b_{\max} . Hence,

$$\begin{aligned} \lambda(a_{\max}, b_{\max}) &\leq \lambda(a, b) \\ &< \min\{d(a), d(b)\} \\ &\leq \min\{d(a_{\max}), d(b_{\max})\}. \end{aligned}$$

□

Property (iii) of pendant trees gives the following lemma.

LEMMA 3.3. *Let AB be an edge of a pendant tree \mathcal{T} and let a_{\max} be a vertex in A of maximum degree. Then $c(AB) < d(a_{\max})$.*

PROOF. Let b_{\max} be a vertex of maximum degree in B and let $a^* \in A$ and $b^* \in B$ be such that $c(AB) = \lambda(a^*, b^*)$ due to Property (iii). By the transitivity of local edge-connectivity, we have

$$\begin{aligned} \lambda(a_{\max}, b_{\max}) &\geq \min\{\lambda(a_{\max}, a^*), \lambda(a^*, b^*), \lambda(b^*, b_{\max})\} \\ &= \min\{d(a^*), \lambda(a^*, b^*), d(b^*)\} \\ &= c(AB), \end{aligned}$$

where the first equality follows from the fact that (a_{\max}, a^*) and (b_{\max}, b^*) are pendant. According to Lemma 3.2, $\lambda(a_{\max}, b_{\max}) < d(a_{\max})$, which gives the claim. □

3.3.1. Constructing Pendant Trees. We will construct a pendant tree by contracting edges in a Gomory-Hu tree. We recall that a *Gomory-Hu tree* of a graph G is a tree on the vertex set $V(G)$ of G . If we replace each vertex v in a Gomory-Hu tree by the singleton $\{v\}$, then it is exactly a cut tree that covers the maximal binary relation. We see a Gomory-Hu tree as such a cut tree.

PROPOSITION 3.4. *Given a Gomory-Hu tree of a graph G , a pendant tree of G can be computed in linear time.*

PROOF. Let \mathcal{T} be a Gomory-Hu tree of G . We see \mathcal{T} as a cut tree covering the maximal irreflexive relation. Throughout the algorithm, we maintain that every pair of distinct vertices that is contained in a node is pendant. Iteratively for every edge AB in \mathcal{T} , we check whether there is a non-pendant pair $\{a, b\}$ with $a \in A$ and $b \in B$. We contract AB in \mathcal{T} and set the

new node as $A \cup B$ if and only if there is no such non-pendant pair. We claim that there is such a non-pendant pair if and only if $\min\{d_G(a_{\max}), d_G(b_{\max})\} > c(AB)$, where a_{\max} and b_{\max} are vertices in A and B with maximum degrees, respectively. The sufficiency follows from Lemma 3.2, and it remains to show that if $\min\{d_G(a_{\max}), d_G(b_{\max})\} \leq c(AB)$, then $\{a, b\}$ is pendant for all $a \in A$ and $b \in B$.

Thus suppose that $\min\{d_G(a_{\max}), d_G(b_{\max})\} \leq c(AB)$. Without loss of generality, let $d_G(a_{\max}) \leq c(AB)$, which implies $d_G(a) \leq c(AB)$ for all $a \in A$. Let $a \in A$ and $b \in B$. By the Gomory-Hu tree properties, \mathcal{T} contains vertices $a^* \in A$ and $b^* \in B$ such that $\lambda_G(a^*, b^*) = c(AB)$; in particular, $d_G(b^*) \geq d_G(a^*) = c(AB)$. Then $\{a, b\}$ is pendant, since

$$\begin{aligned} \lambda_G(a, b) &= \min\{\lambda_G(a, a^*), \lambda_G(a^*, b^*), \lambda_G(b^*, b)\} \\ &= \min\{d_G(a), d_G(a^*), c(AB), d_G(b^*), d_G(b)\} \\ &= \min\{d_G(a), d_G(b)\}. \end{aligned}$$

The first equality is implied by the transitivity of local edge-connectivity, the second by the fact that every vertex pair of a node is pendant, and the third by $d_G(b^*) \geq d_G(a^*) = c(AB) \geq d_G(a)$.

It is not hard to see that the algorithm has a linear running time. \square

Proposition 3.4 implies in particular that every graph has a pendant tree.

The best known running time for a deterministic construction of a Gomory-Hu tree is still based on the classical approach that applies $n - 1$ times the uncrossing technique to find uncrossing cuts on the input graph, and hence in $O(n\theta_{\text{flow}})$, where θ_{flow} is the running time for a maximum flow subroutine (by Dinits' algorithm [Din70, Kar73], $\theta_{\text{flow}} = O(n^{2/3}m)$), where $n := |V(G)|$ and $m := |E(G)|$. For randomized algorithms, Bhalgat et al. [BHKP07] showed that a Gomory-Hu tree of a simple unweighted graph can be constructed in expected running time $\tilde{O}(nm)$, where the tilde hides polylogarithmic factors. Therefore, by our construction above, we conclude that:

COROLLARY 3.5. *Given a simple graph G , a pendant tree of G can be constructed deterministically in running time $O(n^{5/3}m)$, and randomized in expected running time $\tilde{O}(nm)$.*

3.3.2. Large Nodes of Degree 1 and 2. In this section we show that the nodes of a pendant tree have large sizes on average, from which we can derive a lower bound of the number of pendant pairs as every pair of vertices in a node is pendant. For any tree \mathcal{T} whose vertex set partitions $V(G)$, let \mathcal{V}_k be the set of nodes of \mathcal{T} having degree k in \mathcal{T} and let $\mathcal{V}_{>k} := \bigcup_{k'>k} \mathcal{V}_{k'}$. We call the nodes in \mathcal{V}_1 *leaf nodes*. In \mathcal{T} , the set \mathcal{V}_2 induces a family of disjoint paths; we call each such path a *2-path*. We will prove that the leaf nodes of pendant trees as well as the nodes that are contained in 2-paths are large.

LEMMA 3.6. *Let \mathcal{T} be a pendant tree of a simple graph G . Then every leaf node A of \mathcal{T} satisfies $|A| > \delta(G)$.*

PROOF. Let $p := |A| \geq 1$ and let B be the node adjacent to A in \mathcal{T} . By Lemma 3.3, we have $\max_{v \in A} d(v) > c(AB) \geq \sum_{v \in A} (d(v) - (p - 1)) \geq \max_{v \in A} d(v) + \delta(p - 1) - p(p - 1)$, where the last inequality singles out the maximum degree. Therefore, $p > 1$ and $p > \delta$. \square

Let a_{\max} be a vertex of maximal degree in a leaf node A with neighbor B . Since $c(AB) < d(a_{\max})$, A must actually contain a vertex that has all its neighbors in A , as otherwise each of the $d(a_{\max})$ incident edges of a_{\max} would contribute at least one edge to

the edge-cut, either directly or by an incident edge of the corresponding neighbor of a_{\max} . This gives the following corollary of Lemma 3.6, which was first shown by Mader.

COROLLARY 3.7 ([Mad74]). *Let \mathcal{T} be a pendant tree of a simple graph G . Then every leaf node A contains a vertex v with $N(v) \subseteq A$. Hence, every pair in $\{v\} \cup N(v)$ is pendant.*

This already implies that simple graphs contain $\binom{\delta+1}{2} = \Omega(\delta^2)$ pendant pairs. Note that Lemma 3.6 and Corollary 3.7 do not hold for graphs having parallel edges: for example, consider a node A that consists of two vertices of degree δ , which are joined by $\delta - 1$ parallel edges. However, even if the graph is not simple, a leaf node A must always contain at least two vertices due to Lemma 3.3.

COROLLARY 3.8. *Every leaf node of a pendant tree of a graph contains at least two vertices.*

In simple graphs, we thus know that leaf nodes give us a large number of pendant pairs. Since \mathcal{T} is a tree, the number of leaf nodes is completely determined by the number of nodes of degree at least 3, namely $|\mathcal{V}_1| = \sum_{A \in \mathcal{V}_{>2}} (d_{\mathcal{T}}(A) - 2) + 2$. Thus, in order to prove a better lower bound on the number of pendant pairs, we have to consider the case that there are many small nodes of size $o(\delta)$ contained in 2-paths. The following two lemmas prove that (i) for every two adjacent nodes A and B in a 2-path with $|A| + |B| > 2$, we have $|A| + |B| \geq \delta - 1 = \Omega(\delta)$ and (ii) if $\delta(G) \geq 5$ or $\lambda(G) \geq 4$ or $\kappa(G) \geq 3$ and \mathcal{P} is a subpath of a 2-path such that all nodes of \mathcal{P} are singletons, then \mathcal{P} contains at most two nodes. This will be used later to show that the bad situation of many small nodes of size $o(\delta)$ can actually not occur.

LEMMA 3.9. *Let \mathcal{T} be a pendant tree of a simple graph G . Let AB be an edge in \mathcal{T} with $A, B \in \mathcal{V}_2$. If $|A| + |B| > 2$, $|A| + |B| \geq \delta(G) - 1$.*

PROOF. Let $p := |A|$ and $q := |B|$, and let $A'A, BB'$ be edges in \mathcal{T} with $A' \neq B$ and $B' \neq A$. By Lemma 3.3, we have $\sum_{v \in A \cup B} d(v, C_{A'A}) \leq c(A'A) \leq \max_{v \in A} d(v) - 1$ and $\sum_{v \in A \cup B} d(v, C_{B'B}) \leq \max_{v \in B} d(v) - 1$. For $v \in A \cup B$, there are at most $p + q - 1$ edges that are incident to v and $A \cup B$, which implies $d(v, C_{A'A}) + d(v, C_{B'B}) \geq d(v) - (p + q - 1)$ (see Figure 3.1). Therefore,

$$\begin{aligned} & \max_{v \in A} d(v) + \max_{v \in B} d(v) - 2 \\ & \geq \sum_{v \in A \cup B} (d(v, C_{A'A}) + d(v, C_{B'B})) \\ & \geq \sum_{v \in A \cup B} (d(v) - (p + q - 1)) \\ & \geq \max_{v \in A} d(v) + \max_{v \in B} d(v) + (p + q - 2)\delta - (p + q)(p + q - 1), \end{aligned}$$

which gives $(p + q)(p + q - 1) \geq (p + q - 2)\delta + 2$ and thus

$$(p + q)(p + q - 2) \geq (p + q - 2)(\delta - 1).$$

Hence, $p + q \geq \delta - 1$ if $p + q > 2$. □

LEMMA 3.10. *Let \mathcal{T} be a pendant tree of a simple graph G with $|V(\mathcal{T})| > 1$. Let $A = \{v_A\}$ be a node in \mathcal{V}_r with neighborhood $B_1, \dots, B_r \in \mathcal{V}_2$ in \mathcal{T} such that $|A| = |B_1| = \dots = |B_r| = 1$. Let $B'_i \neq A$ be the node that is adjacent to B_i in \mathcal{T} . Then $d(v_A) \leq r^2 - 2\gamma$, where $\gamma :=$*

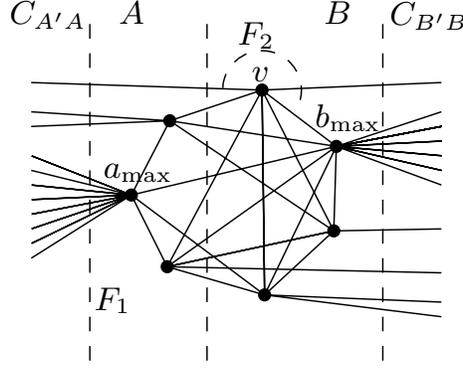


Figure 3.1. A graph G with $\delta = 6$ and adjacent nodes $A, B \in \mathcal{V}_2$ of sizes 3 and 4. Here, $d(a_{\max}) = d(b_{\max}) = 12$, $|F_1| := |c(AA')| = 11 \leq d(a_{\max}) - 1$ and $|F_2| := d(v, C_{A'A}) + d(v, C_{B'B}) = 2 \geq d(v) - (|A| + |B| - 1)$.

$\sum_{1 \leq i < j \leq r} d(C_{B'_i B_i}, C_{B'_j B_j})$. In particular, we have $\delta(G) \leq r^2$ and $\lambda(G) < r^2$. Moreover, if $r = 2$, $\kappa(G) \leq 2$.

PROOF. Note that $r \geq 2$, since there is no singleton leaf node (Corollary 3.8). For every $1 \leq i \leq r$, let $B_i = \{v_i\}$ and $C_i := C_{B'_i B_i}$. Since every v_i or v_A can have at most r neighbors in $\{v_A, v_1, \dots, v_r\}$, we have $d(v_i) \leq r + \sum_{j=1}^r d(v_i, C_j)$ for every $1 \leq i \leq r$, and $d(v_A) \leq r + \sum_{i=1}^r d(v_A, C_i)$. On the other hand, by Lemma 3.3, we have, for every $1 \leq i \leq r$, $d(v_A, C_i) + \sum_{j=1}^r d(v_j, C_i) + \sum_{j \in \{1, \dots, r\} - i} d(C_j, C_i) \leq d(C_i) \leq d(v_i) - 1$ (see Figure 3.2). Therefore,

$$\begin{aligned} & \sum_{i=1}^r \left(d(v_i) + d(v_A, C_i) + \sum_{j=1}^r d(v_j, C_i) + \sum_{j \neq i} d(C_j, C_i) \right) \leq \sum_{i=1}^r \left(r + \sum_{j=1}^r d(v_i, C_j) + d(v_i) - 1 \right) \\ \Leftrightarrow & \sum_{i=1}^r \left(d(v_A, C_i) + \sum_{j \neq i} d(C_j, C_i) \right) \leq r^2 - r \\ \Leftrightarrow & \sum_{i=1}^r d(v_A, C_i) \leq r^2 - r - 2\gamma, \end{aligned}$$

and hence,

$$d(v_A) \leq r + \sum_{i=1}^r d(v_A, C_i) \leq r^2 - 2\gamma.$$

In particular, this gives $\delta(G) \leq r^2$ and, according to Lemma 3.3, $\lambda(G) \leq c(AB_1) < d(v_A) \leq r^2$.

Now, we claim that, if $r = 2$, then $\kappa(G) \leq 2$. If $\gamma(G) > 0$, then $\kappa(G) \leq \delta(G) \leq d(v_A) \leq r^2 - 2\gamma \leq 2$. If $\gamma = 0$, let $S := \{v_A, v_1, v_2\}$, which is a separator of G of size 3. If a vertex $z \in S$ has no neighbor in C_i for some $1 \leq i \leq 2$, $S - z$ is a separator of size 2, which gives the claim. Otherwise, we have $c(B_1 A) \geq 3$ and $c(AB_2) \geq 3$, and in addition, $c(B_1 A) = c(AB_2) = 3$, according to $d(v_A) \leq 4$ and Lemma 3.3. Hence, v_A is of degree 2 in G , which gives the claim. \square

Setting $r = 2$ in Lemma 3.10 gives the following corollary for adjacent nodes of 2-paths.

COROLLARY 3.11. *Let G be simple and let AB and BC be edges in a 2-path of \mathcal{T} . If $\delta(G) \geq 5$ or $\lambda(G) \geq 4$ or $\kappa(G) \geq 3$, then $|A| + |B| + |C| > 3$.*

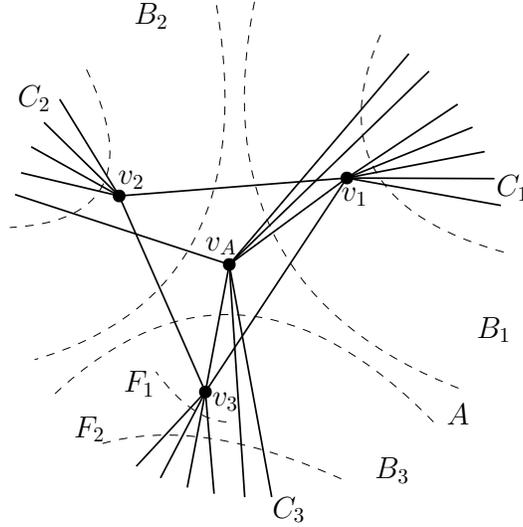


Figure 3.2. A graph with $\delta = 6$. Here, $r = 3$, $|F_1| := \sum_{i=1}^3 d(v_3, C_i) = 4 \leq d(v_3) - r$ and $|F_2| := c(B_3 C_3) = 6 \leq d(v_3) - 1$.

For every node $A \in \mathcal{V}_2$, let A be in $\mathcal{V}_2^{\text{int}}$ if all of its neighbors are also in \mathcal{V}_2 ; otherwise, let A be in $\mathcal{V}_2^{\text{ext}}$. The nodes in $\mathcal{V}_2^{\text{ext}}$ are exactly the endvertices of 2-paths.

LEMMA 3.12. *Let \mathcal{T} be a tree. If $|V(\mathcal{T})| > 1$, then $|\mathcal{V}_{>2}| \leq |\mathcal{V}_1| - 2$ and $|\mathcal{V}_2^{\text{ext}}| \leq 4|\mathcal{V}_1| - 6$.*

PROOF. As \mathcal{T} is a tree, $\sum_{A \in V(\mathcal{T})} d_{\mathcal{T}}(A) = 2|E(\mathcal{T})| = 2(|V(\mathcal{T})| - 1)$, which yields $2 = \sum_{A \in V(\mathcal{T})} (d_{\mathcal{T}}(A) - 2) = \sum_{A \in \mathcal{V}_1} (d_{\mathcal{T}}(A) - 2) + \sum_{A \in \mathcal{V}_2} (d_{\mathcal{T}}(A) - 2) + \sum_{A \in \mathcal{V}_{>2}} (d_{\mathcal{T}}(A) - 2) \geq -|\mathcal{V}_1| + 0 + |\mathcal{V}_{>2}|$, i.e. $|\mathcal{V}_{>2}| \leq |\mathcal{V}_1| - 2$. Since every 2-path contains at most two nodes in $\mathcal{V}_2^{\text{ext}}$ and contracting every 2-path along with one of its neighbors gives a tree \mathcal{T}' with $V(\mathcal{T}') = \mathcal{V}_1 \cup \mathcal{V}_{>2}$, we have $|\mathcal{V}_2^{\text{ext}}| \leq 2E(\mathcal{T}') = 2(|\mathcal{V}_1| + |\mathcal{V}_{>2}| - 1)$. Thus, $|\mathcal{V}_2^{\text{ext}}| \leq 4|\mathcal{V}_1| - 6$. \square

Now we are ready to show that the nodes of 2-paths contain many vertices if $\delta(G) \geq 5$ or $\lambda(G) \geq 4$ or $\kappa(G) \geq 3$.

LEMMA 3.13. *Let \mathcal{T} be a pendant tree of a simple graph G satisfying $\delta(G) \geq 5$ or $\lambda(G) \geq 4$ or $\kappa(G) \geq 3$. Let \mathcal{P} be a 2-path of \mathcal{T} . Then*

$$\sum_{S \in V(\mathcal{P})} |S| \geq (|V(\mathcal{P})| - 2) \frac{\max\{4, \delta(G)\}}{3} + 2.$$

PROOF. For any two consecutive edges AB and BC in \mathcal{P} , applying Corollary 3.11 gives $|A| + |B| + |C| > 3$. Due to Lemma 3.9, this implies $|A| + |B| + |C| \geq \max\{4, \delta\}$. Since there may be at most two singletons that are not contained in such a triple, we conclude $\sum_{S \in V(\mathcal{P})} |S| \geq (|V(\mathcal{P})| - 2) \frac{\max\{4, \delta\}}{3} + 2$. \square

3.3.3. Many Pendant Pairs. We will use the results on large nodes of the previous section to obtain our main theorems, Theorems 3.15 and 3.16. While the latter shows the existence of $\Omega(\delta n)$ pendant pairs, as mentioned in the introduction, the former gives a weaker bound $\Omega(n)$, but in return counts only pendant pairs of a special type.

DEFINITION 3.14. Let a set F of pendant pairs be *dependent* if $V(G)$ contains at least three distinct vertices v_1, \dots, v_k such that $(v_i, v_{i+1}) \in F$ for all $i = 1, \dots, k$, where $v_{k+1} := v_1$; otherwise, F is called *independent*.

Counting only independent pendant pairs allows us to deduce statements about the number of vertices in the graph that is obtained from contracting these pairs (these are not true for arbitrary sets of pendant pairs): Theorem 3.15 will prove for $\delta \geq 5$ that there are at least $\frac{\delta}{\delta+12}n \geq \frac{5}{17}n = \Omega(n)$ such independent pendant pairs. We will show that the contractions imply not only an additive decrease of the number of vertices by at least $\frac{5}{17}n$, but also a multiplicative decrease by the factor δ (i.e. the number of vertices left is $O(n/\delta)$).

THEOREM 3.15. *Let G be a simple graph that satisfies $\delta(G) \geq 5$ or $\lambda(G) \geq 4$ or $\kappa(G) \geq 3$. Let \mathcal{T} be a pendant tree of G . Then G has at least $\frac{\delta(G)|V(G)|}{\delta(G)+12}$ independent pendant pairs each of which is in some node of \mathcal{T} and whose pairwise contraction leaves $O(|V(G)|/\delta(G))$ vertices in the graph.*

PROOF. We may assume that G is connected. Note that $n > \delta \geq 3$ and, for this reason, $\frac{\delta}{\delta+12}n \geq \frac{1}{5}n = \Omega(n)$. First assume that G does not contain a non-pendant pair. For an arbitrary spanning tree of G , consider the pair of endvertices of every edge of it. These are $n - 1 \geq \frac{\delta}{\delta+12}n$ pendant pairs that are independent and whose pairwise contraction leaves only 1 = $O(n/\delta)$ vertex.

Now assume that G contains a non-pendant pair; then $|V(\mathcal{T})| \geq 2$. Using the previous results, we can relate n with the number of nodes in \mathcal{T} by considering the nodes in \mathcal{V}_2 separately as follows.

$$\begin{aligned}
n &= \sum_{S \in V(\mathcal{T})} |S| \\
&= |V(\mathcal{T})| + \sum_{S \in \mathcal{V}_1 \cup \mathcal{V}_{>2}} (|S| - 1) + \sum_{2\text{-path } \mathcal{P}} \left(\sum_{S \in V(\mathcal{P})} |S| - |V(\mathcal{P})| \right) \\
&\geq |V(\mathcal{T})| + |\mathcal{V}_1|\delta + \sum_{2\text{-path } \mathcal{P}, |V(\mathcal{P})| \geq 3} (|V(\mathcal{P})| - 2) \left(\frac{\max\{4, \delta\}}{3} - 1 \right) \\
&\geq |V(\mathcal{T})| + |\mathcal{V}_1|\delta + \frac{1}{12}|\mathcal{V}_2^{\text{int}}|\delta \\
&\geq |V(\mathcal{T})| + \frac{1}{6}(|\mathcal{V}_1| + |\mathcal{V}_2^{\text{ext}}| + |\mathcal{V}_{>2}|)\delta + \frac{1}{12}|\mathcal{V}_2^{\text{int}}|\delta \\
&\geq \left(1 + \frac{1}{12}\delta\right) |V(\mathcal{T})|.
\end{aligned}$$

The first inequality follows from Lemmas 3.6 and 3.13. The second inequality holds, as $\delta = 3 \Rightarrow (\frac{4}{3} - 1) > \frac{\delta}{12}$ and $\delta \geq 4 \Leftrightarrow \frac{\delta}{3} - 1 \geq \frac{\delta}{12}$. The third inequality holds, since $6|\mathcal{V}_1| \geq |\mathcal{V}_1| + |\mathcal{V}_2^{\text{ext}}| + |\mathcal{V}_{>2}|$ follows from Lemma 3.12.

Now let F be any forest on the vertex set $V(G)$ such that, for every $S \in V(\mathcal{T})$, S is the vertex set of a component of F . Then $\{\{u, v\} : uv \in E(F)\}$ is a set of independent pendant pairs. Therefore, G contains at least $|E(F)| = n - |V(\mathcal{T})| \geq (1 - \frac{12}{\delta+12})n = \frac{\delta}{\delta+12}n \geq \frac{n}{5}$ independent pendant pairs. Furthermore, contracting every edge in F leaves at most $|V(\mathcal{T})| \leq (1 + \frac{1}{12}\delta)^{-1}n = O(n/\delta)$ vertices, which gives the second claim. \square

For arbitrary pendant pairs not requiring independence, we improve the lower bound $\Omega(n)$ of Theorem 3.15 to $\Omega(\delta n)$ in the following theorem. This is done by grouping the nodes in a more sophisticated way.

THEOREM 3.16. *Let G be a simple graph that satisfies $\delta(G) \geq 5$ or $\lambda(G) \geq 4$ or $\kappa(G) \geq 3$. Then G contains at least $\frac{\delta(G)|V(G)|}{24}$ pendant pairs.*

PROOF. Note that $n > \delta \geq 3$. If G does not contain a non-pendant pair, there are $\binom{n}{2} \geq \frac{\delta n}{30}$ pendant pairs in G . Otherwise, G contains a non-pendant pair. Let \mathcal{T} be a pendant tree of G ; then $|V(\mathcal{T})| \geq 2$.

For each 2-path \mathcal{P} with $|V(\mathcal{P})| \geq 3$, let \mathcal{P}^* be a subpath obtained from \mathcal{P} by deleting at most two endvertices (i.e. nodes in $\mathcal{P} \cap \mathcal{V}_2^{\text{ext}}$) of \mathcal{P} such that $|V(\mathcal{P}^*)|$ is a multiple of 3. Then, we split \mathcal{P}^* into subpaths $\mathcal{P}_1^*, \dots, \mathcal{P}_{\lfloor \frac{|V(\mathcal{P}^*)|}{3} \rfloor}^*$, each of size 3. By Corollary 3.11 and Lemma 3.9,

$\sum_{S \in V(\mathcal{P}_i^*)} |S| \geq \max\{4, \delta\}$ for every $i = 1, \dots, \lfloor \frac{|V(\mathcal{P}^*)|}{3} \rfloor$. Let $\mathcal{V}_2^* := \mathcal{V}_2 - \bigcup_{2\text{-path } \mathcal{P}, |V(\mathcal{P})| \geq 3} V(\mathcal{P}^*) \subseteq \mathcal{V}_2^{\text{ext}}$. For every leaf node $S \in \mathcal{V}_1$, let Y_S be a collection of nodes that consists of S , at most four nodes from \mathcal{V}_2^* and at most one node from $\mathcal{V}_{>2}$ such that the collections Y_S ($S \in \mathcal{V}_1$) form a partition of $\mathcal{V}_1 \cup \mathcal{V}_2^* \cup \mathcal{V}_{>2}$; such allocation exists as $|\mathcal{V}_2^*| \leq |\mathcal{V}_2^{\text{ext}}| \leq 4|\mathcal{V}_1|$ and $|\mathcal{V}_{>2}| \leq |\mathcal{V}_1|$ (Lemma 3.12). For every $S \in \mathcal{V}_1$, let D_S be a node in Y_S of maximum size. Then, by Lemma 3.6, $|D_S| \geq |S| > \delta$. Thus, the number of pendant pairs in G is at least

$$\begin{aligned}
& \sum_{S \in V(\mathcal{T})} \binom{|S|}{2} \\
& \geq \sum_{S \in \mathcal{V}_1} \frac{|D_S|(|D_S| - 1)}{2} + \sum_{2\text{-path } \mathcal{P}, |V(\mathcal{P})| \geq 3} \sum_{S \in V(\mathcal{P}^*)} \frac{|S|(|S| - 1)}{2} \\
& \geq \frac{\delta}{2} \sum_{S \in \mathcal{V}_1} |D_S| + \frac{1}{2} \sum_{2\text{-path } \mathcal{P}, |V(\mathcal{P})| \geq 3} \sum_{i=1}^{\lfloor \frac{|V(\mathcal{P}^*)|}{3} \rfloor} \sum_{S \in V(\mathcal{P}_i^*)} |S|(|S| - 1) \\
& \geq \frac{\delta}{2} \sum_{S \in \mathcal{V}_1} |D_S| + \frac{1}{2} \sum_{2\text{-path } \mathcal{P}, |V(\mathcal{P})| \geq 3} \sum_{i=1}^{\lfloor \frac{|V(\mathcal{P}^*)|}{3} \rfloor} 3 \left(\frac{\sum_{S \in V(\mathcal{P}_i^*)} |S|}{3} \left(\frac{\sum_{S \in V(\mathcal{P}_i^*)} |S|}{3} - 1 \right) \right) \\
& \geq \frac{\delta}{2} \sum_{S \in \mathcal{V}_1} |D_S| + \frac{\delta}{24} \sum_{2\text{-path } \mathcal{P}, |V(\mathcal{P})| \geq 3} \sum_{S \in V(\mathcal{P}^*)} |S| \\
& \geq \frac{\delta}{12} \sum_{S \in \mathcal{V}_1 \cup \mathcal{V}_2^* \cup \mathcal{V}_{>2}} |S| + \frac{\delta}{24} \sum_{S \in \mathcal{V}_2 - \mathcal{V}_2^*} |S| \\
& \geq \frac{\delta n}{24}.
\end{aligned}$$

The third inequality follows from Jensen's inequality as the function $f(x) := x(x-1)$ is convex for $x \geq 1$. The fourth inequality follows from that $\frac{\sum_{S \in V(\mathcal{P}_i^*)} |S|}{3} - 1 \geq \frac{\delta}{12}$ for every $i = 1, \dots, \lfloor \frac{|V(\mathcal{P}^*)|}{3} \rfloor$, which holds since $\delta \geq 3$ and $\sum_{S \in V(\mathcal{P}_i^*)} |S| \geq \max\{4, \delta\}$ for every $i = 1, \dots, \lfloor \frac{|V(\mathcal{P}^*)|}{3} \rfloor$. □

3.3.4. Tightness. We call a bound *tight* if it is optimal up to a constant factor. Clearly, any graph G contains at most $n - 1$ independent pendant pairs, hence the order of the lower bound in Theorem 3.15 is tight. The order of the number of vertices left after contraction in Theorem 3.15 and that of the number of pendant pairs in Theorem 3.16 are also tight; consider the unions of $\frac{n}{\delta+1}$ many disjoint cliques (i.e. complete subgraphs) $K_{\delta+1}$.

Each of the conditions $\delta(G) \geq 5$, $\lambda(G) \geq 4$ and $\kappa(G) \geq 3$ in Theorems 3.15 and 3.16 is tight, as the graph in Figure 3.3 can be arbitrarily large and satisfies $\delta(G) = 4$, $\lambda(G) = 3$ and $\kappa(G) = 2$ but has only a constant number of pendant pairs. Also the simpleness condition in both results is indispensable: Consider the path graph on n vertices whose two end edges have multiplicity δ and all other edges have multiplicity $\delta/2$. This graph has precisely 2 pendant pairs, each at one of its ends.

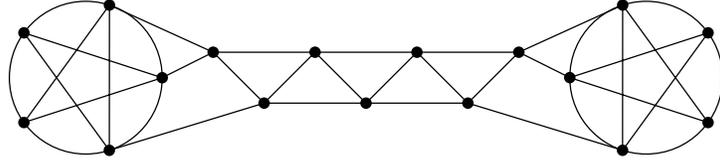


Figure 3.3. The *bone graph* G , whose only pendant pairs are the ones contained in the two K_5 (those form the only leaf nodes of the pendant pair tree). Hence, G has exactly 20 pendant pairs.

3.4. Contraction-Based Sparsification Preserving Small Cuts

In the recent algorithm of Kawarabayashi and Thorup [KT18], a crucial sparsification step is to contract vertex subsets of G such that $O((n \log^c n)/\delta)$ vertices and $O(n \log^c n)$ edges remain for some constant c and all non-trivial min-cuts are preserved, where $n := |V(G)|$. We will show the existence of two such contraction-based sparsifications by considering two cut tree called δ -edge-connectedness tree which covers the pairs $\{v, w\}$ of vertices with $\lambda_G(v, w) < \delta$, and *non-trivial min-cut tree* which covers the pairs of vertices which are separated by some non-trivial min-cut.

By inheriting the argument using large leaf- and \mathcal{V}_2 -nodes for pendant trees to these new cut trees, we will prove that the contraction of every node in these trees leaves only $O(n/\delta)$ vertices and $O(n)$ edges. In this way, all cuts of size less than δ and all non-trivial min-cuts will be preserved, respectively.

3.4.1. The δ -Edge-Connectedness Tree. By definition, every pendant pair of a graph G is $\delta(G)$ -edge-connected. Hence, most of the results about pendant pairs can be transferred directly to statements about δ -edge-connected pairs. In particular, Lemma 3.6 gives the following corollary.

COROLLARY 3.17. *Every simple graph G contains a set S of at least $\delta(G) + 1$ vertices such that $\lambda_G(v, w) \geq \delta(G)$ for every $v, w \in S$.*

More generally, Theorems 3.15 and 3.16 still hold without further ado when we replace the binary pendant pair relation with the δ -edge-connectedness relation on vertex pairs. We now show how these arguments give the first sparsification result described above. To this end, we will use the following relaxation of both Gomory-Hu and pendant trees. A *k -edge-connectedness tree* (also known as *partial Gomory-Hu tree* [BHKP07]) is a cut tree which covers $\{\{v, w\} : \lambda_G(v, w) < k\}$.

A k -edge-connectedness tree \mathcal{T} exists for every graph, as we can contract all edges that induce cuts of size larger than k in a Gomory-Hu tree. Moreover, \mathcal{T} can be computed similarly as in the approach that was used in Proposition 3.4, and hence in deterministic

time $O(n\theta_{\text{flow}})$. For randomized algorithms, [BHKP07] showed that \mathcal{T} can be constructed in expected running time $\tilde{O}(m + nk^2)$, where $m := |E(G)|$.

In this section, we focus on the case that $k = \delta$, i.e. on the δ -edge-connectedness tree \mathcal{T} . By Property (i) of Definition 3.1, every node of \mathcal{T} is δ -edge-connected and therefore a subset of a δ -edge-connected component of G . By Property (ii) and (iii), no δ -edge-connected component intersects two (not necessarily adjacent) nodes A and B of \mathcal{T} , as A and B are separated by a cut of size less than δ . Hence, the nodes of every $\delta(G)$ -edge-connectedness tree are precisely the δ -edge-connected components of G .

3.4.1.1. *Contractions Preserving Small Cuts.* Now we relate \mathcal{T} to any pendant tree \mathcal{T}' of G . Since \mathcal{T}' is pendant, every node of \mathcal{T}' is δ -edge-connected and therefore a subset of some node of \mathcal{T} . Hence, the vertex partition of every pendant tree *refines* the partition of $V(G)$ into δ -edge-connected components. It is also not hard to see that, given a δ -edge-connectedness tree \mathcal{T} , there is a pendant tree \mathcal{T}' , such that contracting all edges $e \in E(\mathcal{T}')$ with $c(e) \geq \delta$ gives \mathcal{T} .

THEOREM 3.18. *Contracting every $\delta(G)$ -edge-connected component of a simple graph G satisfying $\delta(G) > 0$ leaves $O(|V(G)|/\delta(G))$ vertices and $O(|V(G)|)$ edges.*

PROOF. If $1 \leq \delta \leq 4$, then $\delta n \leq 4n$ and hence there are trivially at most $n \leq 4n/\delta = O(n/\delta)$ many vertices left after the contractions. If $\delta \geq 5$, consider any pendant tree \mathcal{T}' of G and contract every node of \mathcal{T}' . Since the partition of any pendant tree refines the partition of $V(G)$ into the δ -edge-connected components of G , Theorem 3.15 implies that these contractions leave only $O(n/\delta)$ vertices.

Now let \mathcal{T} be a δ -edge-connectedness tree of G and consider the edges that are left after the contractions. Every remaining edge is contained in some edge-cut of G that is induced by an edge of \mathcal{T} . Since \mathcal{T} is a δ -edge-connectedness tree, every such edge-cut has size at most $\delta - 1$. By the result above, \mathcal{T} has $O(n/\delta)$ nodes. Hence, there are at most $O(n/\delta) \cdot (\delta - 1) = O(n)$ edges left. \square

Note that the graph after contraction may have multiedges. The following is a fundamental corollary of Theorem 3.18. Despite its generality, it appears to be unknown so far.

COROLLARY 3.19. *Every simple graph G with $\delta(G) > 0$ has $O(|V(G)|/\delta(G))$ many $\delta(G)$ -edge-connected components.*

It was recently shown [LST18] that every simple graph G that satisfies $0 < \lambda(G) < \delta(G)$ has $O((n/\delta)^2)$ min-cuts, we will give a proof of this result in Section 3.4.2. We now strengthen this to cuts of size *not much larger* than $\lambda(G)$ as follows.

THEOREM 3.20. *Every simple graph G that satisfies $0 < \lambda(G) < \delta(G)$ has $O((\frac{|V(G)|}{\delta(G)})^2)$ cuts of size less than $\min\{\frac{3}{2}\lambda(G), \delta(G)\}$.*

PROOF. Henzinger and Williamson [HW96] proved that in any connected graph H the number of cuts of size less than $\frac{3}{2}\lambda(H)$ is at most $O(|V(H)|^2)$. Let G' be the graph obtained from G by contracting every δ -edge-connected component. This preserves every cut of size less than δ . As contractions do not decrease the edge-connectivity of any vertex pair, G' has precisely the same cuts of size less than δ as G . Thus, we can count the number of these cuts in G' instead of in G . Since $\lambda(G') = \lambda(G) > 0$, $\delta(G') > 0$ and G' is connected. Applying Theorem 3.18 to G and [HW96] to $H := G'$ therefore shows that G has $O((n/\delta)^2)$ cuts of size less than $\min\{\frac{3}{2}\lambda, \delta\}$. \square

THEOREM 3.21. *Given any real number $\alpha \geq 1$, every simple graph G that satisfies $0 < \lambda(G) < \delta(G)$ has $O((\frac{|V(G)|}{\delta(G)})^{2\alpha})$ cuts of size at most $\min\{\alpha \cdot \lambda(G), \delta(G) - 1\}$.*

PROOF. Karger [Kar00] proved that in any connected graph H the number of cuts of size at most $\alpha \cdot \lambda(H)$ is in $O(|V(H)|^{2\alpha})$. Again, let G' be the graph obtained from G by contracting every δ -edge-connected component. Applying Theorem 3.18 to G and [Kar00] to $H := G'$ shows that G has $O((n/\delta)^{2\alpha})$ cuts of size at most $\min\{\alpha\lambda, \delta - 1\}$. \square

The same approach can also be used to strengthen various other upper bounds known on the number of small cuts.

3.4.2. The Non-Trivial Min-Cut Tree. Although the $\delta(G)$ -edge-connectedness tree preserves all (not necessarily minimum) cuts of size less than δ , it does not preserve cuts of size δ . However, one cannot expect to preserve all cuts of size δ by contracting vertex subsets within the desired bounds, as the complete graph $K_{\delta+1}$ shows. Hence, we will preserve only non-trivial min-cuts. To this end we consider a new cut tree, which will imply an upper bound on the number of non-trivial min-cuts.

A *non-trivial min-cut tree* \mathcal{T} is a cut tree which covers the pairs of vertices which are separated by some non-trivial min-cuts, and satisfies the following additional property: (iv) for every $AB \in E(\mathcal{T})$, C_{AB} is a non-trivial min-cut. It is clear that Property (iv) implies Properties (ii) and (iii).

Property (i) implies that all non-trivial min-cuts will be preserved if every node is contracted. Property (iv) is equivalent to saying that no leaf node is a singleton and $c(AB) = \lambda$ for all $AB \in E(\mathcal{T})$.

Unlike pendant trees, non-trivial min-cut trees do not exist for every graph. To see this, consider any cycle of length at least four. As every edge is contained in a non-trivial min-cut, every leaf node A of a non-trivial min-cut tree \mathcal{T} is an independent set of size at least two in G . Then the tree edge $AB \in E(\mathcal{T})$ satisfies $c(AB) \geq 4$, which contradicts Property (iv). We conclude that not every graph with $\lambda(G) = 2$ has a non-trivial min-cut tree. However, we will show that non-trivial min-cut trees exist for all simple graphs G with $\lambda(G) \neq 0, 2$.

3.4.2.1. Construct Non-Trivial Min-Cut Tree from Cactus Representation. We call a multigraph \mathcal{K} a *cactus* if it is 2-edge-connected, contains no self-loops, and each edge in \mathcal{K} belongs to exactly one cycle (which may be of length 2, i.e. a pair of parallel edges). This is equivalent to saying that all maximal 2-connected subgraphs of \mathcal{K} are cycles. Note that an edge min-cut in \mathcal{K} is exactly two edges from a cycle in \mathcal{K} . We call vertices of \mathcal{K} *nodes*. Let C be a cycle in \mathcal{K} and v be a node in C . We denote by $\mathcal{K}[C, v]$ the component containing v of the graph obtained from \mathcal{K} by deleting the two edges incident to v in C . We denote by $\mathcal{C}(G)$ the set of all min-cuts of G and by $\mathcal{NC}(G)$ that of all non-trivial min-cuts of G .

A *cactus representation* (\mathcal{K}, φ) of G consists of a cactus \mathcal{K} and a mapping φ from $V(G)$ to $V(\mathcal{K})$ such that (a) for every min-cut X in G , there is a min-cut Y in \mathcal{K} with $X = \varphi^{-1}(Y)$ and (b) for every min-cut Y in \mathcal{K} , $\varphi^{-1}(Y)$ is a min-cut in G . A node v in \mathcal{K} is *empty* if $\varphi^{-1}(v)$ is empty, a *singleton* if $\varphi^{-1}(v)$ consists of exactly one vertex of G , and a *k-junction* if v is contained in exactly k cycles of \mathcal{K} . A cactus representation (\mathcal{K}, φ) of G is *minimal* if one cannot get another cactus representation by contracting an edge of \mathcal{K} and revising the mapping correspondingly.

It has been proven by Dinits et al. [DKL76] that every graph G admits a cactus representation for $\mathcal{C}(G)$. Furthermore, Kawarabayashi and Thorup [KT18] showed that a cactus representation can be computed in near-linear time. We first consider some lemmas

which help us to show that a non-trivial min-cut tree can be constructed from a cactus representation in near-linear time for simple graphs G with $\lambda(G) \neq 0, 2$.

LEMMA 3.22. *Given a cactus representation (\mathcal{K}, φ) of G . Let C be a cycle of length larger than 2 in the cactus \mathcal{K} and let u and v be adjacent nodes in C . Then G has exactly $\lambda(G)/2$ edges between $\varphi^{-1}(\mathcal{K}[C, u])$ and $\varphi^{-1}(\mathcal{K}[C, v])$; in particular, $\lambda(G)$ is even.*

PROOF. Let $X_1 := \varphi^{-1}(\mathcal{K}[C, u])$, $X_2 := \varphi^{-1}(\mathcal{K}[C, v])$ and $X_3 := V - X_1 - X_2$. As (\mathcal{K}, φ) is a cactus representation, X_1 , X_2 and X_3 are min-cuts in G , respectively. This implies that $d(X_1, X_2) + d(X_1, X_3) = d(X_2, X_3) + d(X_2, X_1) = d(X_3, X_1) + d(X_3, X_2) = \lambda(G)$. Therefore, $d(X_1, X_2) = \lambda(G)/2$. \square

PROPOSITION 3.23. *Let G be a simple graph with $\lambda(G) \neq 0, 2$. Then a non-trivial min-cut tree \mathcal{T} of G can be computed in time $\tilde{O}(|E(G)|)$.*

PROOF. Let (\mathcal{K}, φ) be a minimal cactus representation of G . Note that all min-cuts of G are represented by the min-cuts of \mathcal{K} . We can simply use $|V(\mathcal{K})| - 1$ uncrossing min-cuts (each of them constitutes of exactly two edges in some cactus cycle) to split the cactus nodes, from which a cut tree will result. The edges of this tree will represent some but possibly not all min-cuts of G , nevertheless, all min-cuts of G will be preserved if we contract the vertex sets represented by the tree nodes. We will choose these $|V(\mathcal{K})| - 1$ uncrossing min-cuts of \mathcal{K} carefully for each cycle in \mathcal{K} in order to fulfill the Properties (i) and (iv).

Let C be a cycle $v_1 v_2 \dots v_l v_1$ of length l in \mathcal{K} . If there are two distinct cactus nodes say v_1 and v_i ($1 < i \leq l$) in C with $|\varphi^{-1}(\mathcal{K}[C, v_1])| > 1$ and $|\varphi^{-1}(\mathcal{K}[C, v_i])| > 1$. We split \mathcal{K} with the $l - 1$ cuts $\{v_1, v_2, \dots, v_j\}$ (for $j = 1, \dots, i - 1$) and $\{v_i, v_{i+1}, \dots, v_j\}$ (for $j = i, \dots, l - 1$). Each of these $l - 1$ cuts represents some non-trivial min-cut of G .

If, for every cactus node v in C except v_1 , we have $|\varphi^{-1}(\mathcal{K}[C, v])| = 1$, we claim that $l = 2$. Otherwise, if $l > 2$, then by Lemma 3.22, λ must be even. By our condition $\lambda \neq 0, 2$, we have $\lambda \geq 4$. By Lemma 3.22, there are $\lambda/2 \geq 2$ edges between $\varphi^{-1}(\mathcal{K}[C, v_2])$ and $\varphi^{-1}(\mathcal{K}[C, v_3])$, which is not possible since $|\varphi^{-1}(\mathcal{K}[C, v_2])| = |\varphi^{-1}(\mathcal{K}[C, v_3])| = 1$ and G is simple. We split this cycle of length 2 with the only possible cut. By the minimality of (\mathcal{K}, φ) , we know that the cactus node v_2 is a 1-junction singleton in C and all non-trivial min-cuts will be preserved if we contract the vertex set $\varphi^{-1}(v_1) \cup \varphi^{-1}(v_2)$.

We now have a tree on $V(\mathcal{K})$ obtained by splitting the cycles in \mathcal{K} . The cuts represented by the tree edges are non-trivial min-cuts in G , except those obtained from some cycle of length 2 and containing 1-junction singleton. As we discussed before, those tree edges can be simply contracted. We also contract it if there is an edge which has an empty node as its endvertex. It is clear that after these modifications all non-trivial min-cuts are still preserved, the tree nodes form a partition of $V(G)$ and all cuts represented by the tree edges are non-trivial. Therefore we obtain a non-trivial min-cut tree. We use the result of Kawarabayashi and Thorup [KT18] to find a cactus representation of G in near-linear time. All the steps afterwards including verifying the minimality of (\mathcal{K}, φ) , searching cuts for each cactus cycle and contracting several tree edges can be done in linear time. We conclude that a non-trivial min-cut tree can be constructed in near-linear time. \square

3.4.2.2. Contractions Preserving Non-Trivial Min-Cuts. The following lemma assures that leaf nodes must have size $\Omega(\delta)$.

LEMMA 3.24. *Every non-trivial min-cut $A \subset V(G)$ of a simple graph G satisfies $|A| \geq \delta(G)$. In particular, every leaf node A of a non-trivial min-cut tree of G satisfies $|A| \geq \delta(G)$.*

PROOF. For the first claim, let $p := |A|$. Then $\delta \geq \lambda \geq \sum_{v \in A} (d(v) - (p-1)) \geq p\delta - p(p-1)$ implies $p \geq \delta$, as $p > 1$. The second claim follows directly from the first. \square

We remark that cut tree covering pairs of vertices separated by non-trivial min-cut also exists for graph with $\lambda = 0, 2$ if we do not require Property (iv) in addition, but then Lemma 3.24 does not always hold for such cut tree.

The following analogues of Lemmas 3.9 and 3.10 will ensure that the number of vertices will decrease by a factor of $\Omega(\delta)$ when contracting all nodes of a non-trivial min-cut tree.

LEMMA 3.25. *Let \mathcal{T} be a non-trivial min-cut tree of a simple graph G . Let $A'A, AB, BB'$ be edges in \mathcal{T} such that $A, B \in \mathcal{V}_2$. If $|A| + |B| > 2$, $|A| + |B| \geq \delta(G)/2$.*

PROOF. Let $p := |A|$ and $q := |B|$. It is clear that $\sum_{v \in A \cup B} d(v, C_{A'A}) \leq \lambda \leq \delta$, $\sum_{v \in A \cup B} d(v, C_{B'B}) \leq \delta$ and $d(v, C_{A'A}) + d(v, C_{B'B}) \geq d(v) - (p + q - 1)$. Therefore, $2\delta \geq \sum_{v \in A \cup B} (d(v, C_{A'A}) + d(v, C_{B'B})) \geq \sum_{v \in A \cup B} (d(v) - (p + q - 1)) \geq (p + q)(\delta - (p + q - 1))$, which gives $p + q \geq \frac{p+q-2}{p+q-1} \cdot \delta \geq \frac{1}{2} \cdot \delta$ if we assume $p + q > 2$. \square

LEMMA 3.26. *Let \mathcal{T} be a non-trivial min-cut tree of a simple graph G . Let A be a node in \mathcal{V}_r with neighborhood $B_1, \dots, B_r \in \mathcal{V}_2$ in \mathcal{T} such that $|A| = |B_1| = \dots = |B_r| = 1$. Then $\delta(G) \leq r^2 + r$.*

PROOF. Let $A := \{v_A\}$. For every $1 \leq i \leq r$, let $B_i := \{v_i\}$ and $B'_i \neq A$ be the node that is adjacent to B_i in \mathcal{T} . Write $C_i := C_{B'_i B_i}$. Since every v_i or v_A can have at most r neighbors in $\{v_A, v_1, \dots, v_r\}$, we have, for every $1 \leq i \leq r$, $d(v_i) \leq r + \sum_{j=1}^r d(v_i, C_j)$, and $d(v_A) \leq r + \sum_{i=1}^r d(v_A, C_i)$. On the other hand, we have, for every $1 \leq i \leq r$, $d(v_A, C_i) + \sum_{j=1}^r d(v_j, C_i) \leq \lambda \leq \delta$. Therefore, $\sum_{i=1}^r (\delta + d(v_A, C_i) + \sum_{j=1}^r d(v_j, C_i)) \leq \sum_{i=1}^r (d(v_i) + d(v_A, C_i) + \sum_{j=1}^r d(v_j, C_i)) \leq \sum_{i=1}^r (r + \sum_{j=1}^r d(v_i, C_j) + \delta)$, which implies $r^2 \geq \sum_{i=1}^r d(v_A, C_i) \geq d(v_A) - r \geq \delta - r$. \square

Now we present an alternative proof of the sparsification result in [LST18].

THEOREM 3.27. *Let G be a simple graph with $\delta(G) > 0$ and let \mathcal{T} be a non-trivial min-cut tree of G . Then contracting every node of \mathcal{T} leaves $O(|V(G)|/\delta(G))$ vertices and $O(|V(G)|)$ edges.*

PROOF. We can assume $\delta \geq 7$, as otherwise $\delta n \leq 6n$, which implies that there are at most $n \leq 6n/\delta = O(n/\delta)$ vertices left after the contractions. We can also assume that $|V(\mathcal{T})| > 1$, as otherwise the contraction leaves precisely $1 = O(n/\delta)$ vertex; in particular, we have $\mathcal{V}_0 = \emptyset$. Since $\delta \geq 7$, Lemma 3.26 implies that there are no distinct nodes $B_1, B_2, B_3 \in \mathcal{V}_2$ satisfying $B_1 B_2, B_2 B_3 \in E(\mathcal{T})$. We conclude by Lemma 3.25 that, for every 2-path \mathcal{P} , $\sum_{S \in \mathcal{V}_2^{\text{int}} \cap V(\mathcal{P})} |S| = |\mathcal{V}_2^{\text{int}} \cap V(\mathcal{P})| \cdot \Omega(\delta)$. By Lemmas 3.24 and 3.12, the number of vertices

can be bounded as follows.

$$\begin{aligned}
n &= \sum_{S \in V(\mathcal{T})} |S| \\
&\geq \sum_{S \in \mathcal{V}_1 \cup \mathcal{V}_{>2}} |S| + \sum_{2\text{-path } \mathcal{P}} \left(\sum_{S \in \mathcal{V}_2^{\text{int}} \cap V(\mathcal{P})} |S| \right) \\
&\geq |\mathcal{V}_1| \cdot \Omega(\delta) + \sum_{2\text{-path } \mathcal{P}} (|\mathcal{V}_2^{\text{int}} \cap V(\mathcal{P})| \cdot \Omega(\delta)) \\
&= (|\mathcal{V}_1| + |\mathcal{V}_2^{\text{ext}}| + |\mathcal{V}_{>2}|) \cdot \Omega(\delta) + |\mathcal{V}_2^{\text{int}}| \cdot \Omega(\delta) \\
&= |V(\mathcal{T})| \cdot \Omega(\delta).
\end{aligned}$$

Therefore, $|V(\mathcal{T})| = O(n/\delta)$ vertices and at most $(|V(\mathcal{T})| - 1) \cdot \lambda = O(n\lambda/\delta) \leq O(n)$ edges will be left if all nodes of \mathcal{T} are contracted. \square

3.4.3. Tightness. We show that the above results are tight, except for the cases in which this was already shown. The following graph shows that the bounds of Corollaries 3.17 and 3.19 and Theorems 3.18 and 3.27 (vertex- and edge-bound) are tight. Let $n \geq 3(\delta + 1)$, $\delta \geq 2$ and assume that n is a multiple of $\delta + 1$ (the last assumption can be avoided by a simple modification of the construction). Then the graph G obtained from the cycle on $n/(\delta + 1)$ vertices by replacing all vertices with a copy of $K_{\delta+1}$ shows tightness.

Although this fixes $\lambda(G) = 2$, this example can be readily generalized to tight graphs having larger and even λ such that $\lambda < \delta/2$. To do so, obtain a graph G' from G by adding $\lambda/2 - 1$ cycles that are vertex-disjoint from the first initial cycle C , but visit exactly the same complete subgraphs in the same order as C .

Bibliography

- [AMLM13] E. Álvarez Miranda, I. Ljubić, and P. Mutzel, *The maximum weight connected subgraph problem*, Facets of Combinatorial Optimization (M. Jünger and G. Reinelt, eds.), Springer, Berlin, 2013, p. 245–270.
- [BHKP07] A. Bhalgat, R. Hariharan, T. Kavitha, and D. Panigrahi, *An $\tilde{O}(mn)$ Gomory-Hu tree construction algorithm for unweighted graphs*, Proceedings of the 39th Annual Symposium on Theory of Computing (STOC’07), 2007, pp. 605–614.
- [Bon75] J. A. Bondy, *Pancyclic graphs: Recent results*, Infinite and finite sets, Vol. I (Colloq. Keszthely, 1973; Dedicated to P. Erdős on his 60th birthday) (A. Hajnal, R. Rado, and V.T. Sós, eds.), Colloq. Math. Soc. János Bolyai, vol. 10, North-Holland, Amsterdam, 1975, pp. 181–187.
- [Cai93] M.-C. Cai, *The number of vertices of degree k in a minimally k -edge-connected graph*, J. Combin. Theory Ser. B **58** (1993), no. 2, 225–239.
- [CFY04] G. Chen, G. Fan, and X. Yu, *Cycles in 4-connected planar graphs*, European J. Combin. **25** (2004), 763–780.
- [Cho77] S. A. Choudum, *Some 4-valent, 3-connected, planar, almost pancyclic graphs*, Discrete Math. **18** (1977), no. 2, 125–129.
- [CHW09] Q. Cui, Y. Hu, and J. Wang, *Long cycles in 4-connected planar graphs*, Discrete Math. **309** (2009), no. 5, 1051–1059.
- [CN89] N. Chiba and T. Nishizeki, *The hamiltonian cycle problem is linear-time solvable for 4-connected planar graphs*, J. Algorithms **10** (1989), no. 2, 187–211.
- [CNAO85] N. Chiba, T. Nishizeki, S. Abe, and T. Ozawa, *A linear algorithm for embedding planar graphs using PQ-trees*, J. Comput. System Sci. **30** (1985), no. 1, 54–76.
- [Din70] E. A. Dinic, *Algorithm for solution of a problem of maximum flow in a network with power estimation*, Soviet Math Doklady **11** (1970), 1277–1280.
- [DKL76] E. A. Dinitz, A. V. Karzanov, and M. V. Lomonosov, *On the structure of a family of minimal weighted cuts in a graph*, Studies in Discrete Optimization (in Russian) (Nauka, Moscow) (A. A. Fridman, ed.), 1976, pp. 290–306.
- [FJMv02] G. Fijavž, M. Juvan, B. Mohar, and R. Škrekovski, *Planar graphs without cycles of specific lengths*, European J. Combin. **23** (2002), no. 4, 377–388.
- [Fra94] A. Frank, *On the edge-connectivity algorithm of Nagamochi and Ibaraki*, Laboratoire Artemis, IMAG, Université J. Fourier, Grenoble, 1994.
- [Gar05] N. Garg, *Saving an epsilon: a 2-approximation for the k -MST problem in graphs*, Proceedings of the 37th Annual ACM Symposium on Theory of Computing (STOC’05), 2005, pp. 396–402.
- [Hie73] C. Hierholzer, *Ueber die Möglichkeit, einen Linienzug ohne Wiederholung und ohne Unterbrechung zu umfahren*, Math. Ann. **6** (1873), no. 1, 30–32.
- [HK08] M. Hornák and Z. Kocková, *On planar graphs arbitrarily decomposable into closed trails*, Graphs Combin. **24** (2008), no. 1, 19–28.
- [HRW17] M. Henzinger, S. Rao, and D. Wang, *Local flow partitioning for faster edge connectivity*, Proceedings of the 28th Annual Symposium on Discrete Algorithms (SODA’17), 2017, pp. 1919–1938.
- [HW96] M. Henzinger and D. P. Williamson, *On the number of small cuts in a graph*, Inform. Process. Lett. **59** (1996), 41–44.
- [INS⁺12] T. Ito, T. Nishizeki, M. Schröder, T. Uno, and X. Zhou, *Partitioning a weighted tree into subtrees with weights in a given range*, Algorithmica **62** (2012), no. 3–4, 823–841.
- [Kar73] A. V. Karzanov, *On finding a maximum flow in a network with special structure and some applications*, Matematicheskie Voprosy Upravleniya Proizvodstvom (in Russian) (1973), 81–94.

- [Kar00] D. R. Karger, *Minimum cuts in near-linear time*, J. ACM **47** (2000), no. 1, 46–76.
- [KT18] K. Kawarabayashi and M. Thorup, *Deterministic edge connectivity in near-linear time*, J. ACM **66** (2018), no. 1, 4:1–4:50.
- [LS18] O.-H. S. Lo and J. M. Schmidt, *A cut tree representation for pendant pairs*, Proceedings of the 29th International Symposium on Algorithms and Computation (ISAAC'18), 2018, pp. 38:1–38:9.
- [LST18] O.-H. S. Lo, J. M. Schmidt, and M. Thorup, *Contraction-based sparsification in near-linear time*, Manuscript available online at www.arxiv.org/abs/1810.03865, 2018.
- [Mad71] W. Mader, *Existenz gewisser Konfigurationen in n -gesättigten Graphen und in Graphen genügend großer Kantendichte*, Math. Ann. **194** (1971), 295–312.
- [Mad73] ———, *Grad und lokaler Zusammenhang in endlichen Graphen*, Math. Ann. **205** (1973), 9–11.
- [Mad74] ———, *Kantendisjunkte Wege in Graphen*, Monatsh. Math. **78** (1974), no. 5, 395–404.
- [Mad95] ———, *On vertices of degree n in minimally n -edge-connected graphs*, Combin. Probab. Comput. **4** (1995), no. 1, 81–95.
- [Mad96] ———, *On vertices of degree n in minimally n -connected graphs and digraphs*, Combinatorics, Paul Erdős is eighty, Vol. 2 (Keszthely, 1993) (D. Miklós, V.T. Sós, and T. Szőnyi, eds.), Bolyai Soc. Math. Stud., vol. 2, János Bolyai Math. Soc., Budapest, 1996, pp. 423–449.
- [Mal71] J. Malkevitch, *On the lengths of cycles in planar graphs*, Recent Trends in Graph Theory (Proc. Conf., New York 1970) (M. Capobianco, J.B. Frechen, and M. Krolík, eds.), Lecture Notes in Mathematics, vol. 186, Springer, Berlin, 1971, p. 191–195.
- [Mal78] ———, *Cycle lengths in polytopal graphs*, Theory and Applications of Graphs (Proc. Conf., Michigan 1976) (Y. Alavi and D.R. Lick, eds.), Lecture Notes in Computer Science, vol. 642, Springer, Berlin, 1978, p. 364–370.
- [Mal88] ———, *Polytopal graphs*, Selected Topics in Graph Theory (L. Beineke and R. Wilson, eds.), vol. 3, Academic Press, 1988, p. 169–188.
- [Moh18] S. Mohr, *Personal communication*, 2018.
- [MT] T. Madaras and M. Tamášová, *Minimal unavoidable sets of cycles in plane graphs with restricted minimum degree and edge weight*, Manuscript.
- [NI92] H. Nagamochi and T. Ibaraki, *Computing edge-connectivity in multigraphs and capacitated graphs*, SIAM J. Discrete Math. **5** (1992), no. 1, 54–66.
- [Plu75] M. D. Plummer, *Problems*, Infinite and finite sets, Vol. III (Colloq. Keszthely, 1973; Dedicated to P. Erdős on his 60th birthday) (A. Hajnal, R. Rado, and V.T. Sós, eds.), Colloq. Math. Soc. János Bolyai, vol. 10, North-Holland, Amsterdam, 1975, pp. 1549–1550.
- [San97] D. P. Sanders, *On paths in planar graphs*, J. Graph Theory **24** (1997), 341–345.
- [SW97] M. Stoer and F. Wagner, *A simple min-cut algorithm*, J. ACM **44** (1997), no. 4, 585–591.
- [Tho83] C. Thomassen, *A theorem on paths in planar graphs*, J. Graph Theory **7** (1983), no. 2, 169–176.
- [Tre89] M. Trenkler, *On 4-connected, planar 4-almost pancyclic graphs*, Math. Slovaca **39** (1989), no. 1, 13–20.
- [Tut56] W. T. Tutte, *A theorem on planar graphs*, Trans. Amer. Math. Soc. **82** (1956), 99–116.
- [TY84] R. E. Tarjan and M. Yannakakis, *Simple linear-time algorithms to test chordality of graphs, test acyclicity of hypergraphs, and selectively reduce acyclic hypergraphs*, SIAM J. Comput. **13** (1984), no. 3, 566–579.
- [TY94] R. Thomas and X. Yu, *4-connected projective-planar graphs are hamiltonian*, J. Combin. Theory Ser. B **62** (1994), 114–132.
- [Vyg11] J. Vygen, *Splitting trees at vertices*, Discrete Math. **311** (2011), no. 1, 67–69.
- [WL02] W. Wang and K.-W. Lih, *Choosability and edge choosability of planar graphs without 5-cycles*, Appl. Math. Lett. **15** (2002), no. 5, 561–565.

Symbol Index

- (S, l, M, n) , 6
 (\mathcal{K}, φ) , 34
 $A - B$, vii
 C , 14
 C_T , 4
 C_{AM} , 24
 C_{ext} , 14
 C_{int} , 14
 $E(G)$, vii, viii
 $E_G(X, Y), E_G(v, Y)$, vii
 $G - F$, vii
 $G - W$, vii
 $G[W]$, vii
 G^* , viii
 G_{ext} , 14
 G_{int} , 14
 $N_G(v)$, vii
 $Q_{u,v}$, 7
 R , vii, 23
 $T[vw; v]$, viii
 T^* , 7
 $T^*[\tilde{r}]$, 7
 $T^{(a)}$, viii
 $T_v^{(a)}$, viii
 $V(G)$, vii, viii
 $V_i(T), V_i$, 5
 $[u, v]_C, [u, v]$, viii
 \mathcal{T} , 24
 $\mathcal{V}_2^{\text{ext}}$, 29
 $\mathcal{V}_2^{\text{int}}$, 29
 \mathcal{V}_k , 26
 $\mathcal{V}_{>k}$, 26
 $\delta(G), \delta$, vii
 $\kappa(G), \kappa$, viii
 $\lambda(G), \lambda$, viii
 $\lambda_G(u, v)$, viii
 $[x]$, vii
 $\lfloor x \rfloor$, vii
 $\mathcal{C}(G)$, 34
 \mathcal{K} , 34
 $\mathcal{K}[C, v]$, 34
 $\mathcal{NC}(G)$, 34
 $\mathcal{Q}(T; c, k), \mathcal{Q}$, 7
 \overline{X} , vii
 $\rho([u, v])$, 4
 $\rho(w_{v,i})$, 4
 θ_{flow} , 26
 $|X|$, vii
 $|x|$, vii
 $c(AB)$, 24
 $c([u, v])$, 4
 $c(v)$, 4, 14
 $d_G(X), d_G(v)$, vii
 $d_G(X, Y), d_G(v, Y)$, vii
 $e_{v,i}$, 4
 $f = O(g)$, vii
 $f = \Omega(g)$, vii
 $f = o(g)$, vii
 u^+ , viii
 uv , vii, viii
 v^- , viii
 $v_1v_2 \dots v_tv_1$, vii
 $v_1v_2 \dots v_t$, vii
 $w_{v,i}$, 4
 $CS(G)$, 13

Index

- i*-triangle, 17
- k*-edge-connected, viii
 - component, viii
- adjacent
 - faces, viii
 - vertices, vii
- binary relation, vii, 23
 - irreflexive, vii
 - symmetric, vii
- cactus, 34
 - representation, 34
 - minimal, 34
- connected
 - component, vii
 - graph, viii
- contract, viii
- cut, viii
 - u-v*-, viii
 - minimum, viii
 - min-, viii
 - trivial, viii
- cycle, vii
 - directed, viii
 - Hamilton, 14
 - square, 17
- cycle spectrum, 13
- degree, vii
 - minimum, vii
- discharge, 6
 - last, 6
- dual graph, viii
- edge, vii
 - directed, viii
- edge set, vii, viii
- edge-connectivity, viii
- endvertex
 - edge, vii
 - path, vii
- Euler tour technique, 4
- face, viii
- forest, viii
- graph
 - k*-connected, viii
 - bone, 32
 - complete, viii
 - directed, viii
 - hamiltonian, 13
 - planar, viii
 - plane, viii
 - simple, viii
 - undirected, vii
- incident, vii
- leaf, viii
- length
 - cycle, vii
 - path, vii
- Mohr's transformation, 14, 15
- multigraph, viii
- neighbor, vii
- node, 24, 34
 - k*-junction, 34
 - empty, 34
 - leaf, 26
 - singleton, 34
- overload, 6
- overload-discharge process, 6
- overload-discharge quadruple, 6
 - associated with *u, v*, 7
 - maximal, 7
- overloading, 5
 - k*-overloading, 5
 - path, 5
 - subtree, 5
- pancyclic, 13
 - almost, 13
- path, vii
 - 2-, 26
 - directed, viii
- pendant pair, 23

- pendant pairs
 - dependent, 29
 - independent, 29
- planar embedding, viii

- separate, viii
- singleton, vii
- size
 - cut, viii
 - path, vii
 - set, vii
- subgraph, vii
 - induced, vii
 - spanning, vii
- subtree, viii
- support
 - subtree, 7
 - vertex, 7

- tree, viii
 - k -edge-connectedness, 32
 - cut, 24
 - Gomory-Hu, 24, 25
 - non-trivial min-cut, 34
 - pendant, 25

- vertex, vii
- vertex set, vii, viii
- vertex-connectivity, viii