Product Structure, Separating Systems, Freeze-Tag Problem, and Planar Multicolor Turán Number

Saeed Odak

Thesis submitted to the University of Ottawa
in partial fulfilment of the requirements for
the degree of Doctor of Philosophy in
Computer Science.

School of Electrical Engineering and Computer Science
Faculty of Engineering
University of Ottawa

Preface

This integrated thesis consists of four chapters. Three of these chapters correspond to articles that have already undergone peer review and have been published, as listed below:

- Prosenjit Bose, Pat Morin, and Saeed Odak,
  An Optimal Algorithm for Product Structure in Planar Graphs,
  18th Scandinavian Symposium and Workshops on Algorithm Theory (SWAT).
- Ahmad Biniaz, Prosenjit Bose, Jean-Lou De Carufel, Anil Maheshwari, Babak Miraftab, Saeed Odak, Michiel Smid, Shakhar Smorodinsky, and Yelena Yuditsky,
  On Separating Path and Tree Systems in Graphs,
  Journal of Discrete Mathematics & Theoretical Computer Science, vol. 27:2.
- Nicolas Bonichon, Cyril Gavoille, Nicolas Hanusse, and Saeed Odak,
  Euclidean Freeze-Tag Problem on the Plane,
  36th Canadian Conference on Computational Geometry (CCCG).

The fourth chapter has yet to be developed into a publication:

- Prosenjit Bose, Vida Dujmović, Pat Morin, and Saeed Odak,
  On Rainbow Turán Triangles in Planar Graphs.

As is common in theoretical computer science, the authors are listed alphabetically by family name. While it is often difficult to attribute specific ideas to individual contributors in multi-author papers, I have made substantial contributions that were central to the results presented in each of these three published articles.

Abstract

This thesis is based on a collection of papers focused on graph theory and computational geometry.

The first paper focuses on the *Product Structure Theorem* for planar graphs, which asserts that any planar graph can be embedded in the strong product of a planar 3-tree, a path, and a 3-cycle. The paper presents a simple linear-time algorithm to find this decomposition for an n-vertex planar graph, improving on the previous $O(n \log n)$ time algorithm.

In the second paper, the concept of *vertex-separating systems* is explored, where a separating system is a collection of vertex subsets that is used to distinguish between any two distinct elements of the vertex set. The paper investigates the minimum size of vertex-separating path and tree systems for various types of graphs, including trees, grids, and maximal outerplanar graphs.

The third paper tackles the *geometric freeze-tag problem*, an optimization problem where the goal is to minimize the total wake-up time for a swarm of robots starting with a single active robot. The authors prove a conjecture by Bonichon et al. regarding an upper bound on the wake-up time for robots in convex position and provide an upper bound of $4.63r$ for robots located in a disk of radius r in the ℓ_2-norm, improving the best known bound of $5\sqrt{2}r \approx 7.07r$.

Finally, the fourth paper addresses the *planar rainbow Turán problem*, where the objective is to determine the smallest number of maximal planar graphs required on the same set of n vertices to guarantee a rainbow triangle (a triangle formed by edges from three distinct graphs). The paper establishes both upper and lower bounds, showing that a sequence of at least $0.75n$ planar graphs guarantees a rainbow triangle, and provides an almost linear lower bound using Behrend's construction for three-term arithmetic progression-free sets.

Together, these works contribute to both theoretical and algorithmic advancements in graph decompositions, separating systems, and optimization problems within geometric and graph theoretic settings.

Acknowledgements

First off, big love to my parents — Arkadagym atama we ynam çeşmesi ejeme tüýs ýürekden minnetdarlyk bildirýärin. Your endless love, support, and guidance mean the world to me. I couldn't have done this without your light.

Shoutout to my academic supervisors: Michiel Smid, Anil Maheshwari, Prosenjit Bose, Pat Morin, Jean-Lou De Carufel, and Vida Dujmović at the Algorithms, Graphs, and Geometry Lab at Carleton. Y'all seriously rock. Thanks for shaping my brain and pushing me to do better.

Special thanks to Pat for always coming through with answers, and to Anil and Michiel for the after-lunch meetings and tea times — those discussions were essential.

To my amazing friends — Mohammad Mahdi, Ali, Alma, Hussein, and Mariami — thank you for the laughs, the deep conversations, and the good vibes. Y'all kept me sane.

Much love and gratitude to everyone who's been part of this wild ride.

Contents

List of Tables

List of Figures

Chapter 0

Introduction

In this thesis, we presents a various problems in graph theory and computational geometry. Throughout this thesis, we use standard graph theory terminology as used in the textbook by Diestel [26]. All graphs discussed here are simple, finite and undirected. For a graph G, $V(G)$ and $E(G)$ denote the vertex and edge sets of G, respectively.

Each chapter of the thesis is centered on a distinct published or ongoing research problem. Below, we provide an overview of each chapter along with the relevant results. Every chapter in this thesis includes the definition of the primary research problem and a literature review in its corresponding introduction section. Subsequently, we outline our contribution in a dedicated section and conclude with a list of open questions left for future directions. The contents of the chapters are as follows:

Chapter 1: An Optimal Algorithm for Product Structure in Planar Graphs

For two graphs G_1 and G_2, the *strong graph product* of G_1 and G_2, denoted by $G_1 \boxtimes G_2$, is a graph whose vertex set is $V(G_1 \boxtimes G_2) := V(G_1) \times V(G_2)$ and that contains an edge between distinct vertices $v = (v_1, v_2)$ and $w = (w_1, w_2)$ if and only if (i) $v_1 = w_1$ and $v_2w_2 \in E(G_2)$; (ii) $v_2 = w_2$ and $v_1w_1 \in E(G_1)$; or (iii) $v_1w_1 \in E(G_1)$ and $v_2w_2 \in E(G_2)$.

The *Product Structure Theorem* for planar graphs by Dujmović et al. [27] states that any planar graph G is contained in the strong product of a planar 3-tree, a path, and a 3-cycle. That is, $G \subseteq H \boxtimes P \boxtimes K_3$, where H is a planar 3-tree, P is a path and K_3 is a cycle on 3 vertices. Recall that a planar 3-tree is a graph that can be constructed starting from K_3 and recursively adding vertices such that each new vertex is adjacent to all three vertices of a triangular face in the existing graph, while maintaining planarity. Note that, usually 3-trees compared to the original graph G has a simpler structure. Intuitively, the product structure theorem states that every planar graph G is isomorphic to a subgraph of the product of two or more "simple" graphs.

In this chapter, we give a simple linear-time algorithm for finding this decomposition as well as several related decompositions for given n-vertex planar graph. This improves on the previous $O(n \log n)$ time algorithm by Morin [54].

This theorem and its variants have recently been used as a key tool in resolving several longstanding open problems on planar graphs [18, 25, 27, 29, 30, 32]. For example, in $O(n)$-time for any n-vertex planar graph G, one can compute an $O(1)$-queue layout of G [27]; assign $(1+o(1))\log n$-bit labels to the vertices of G so that one can determine from the labels of vertices v and w whether or not v and w are adjacent in G [29]; and colour the vertices of G with $O(\log n/\log\log\log n)$ integers so that the maximum colour that appears on any path P of length at most ℓ appears at exactly one vertex of P (for any fixed $\ell \geq 2$) [18].

This chapter of the thesis is joint work with Prosenjit Bose and Pat Morin and appeared in the proceedings of the *Scandinavian Symposium and Workshops on Algorithm Theory (SWAT 2022)* [19].

Chapter 2: Separating Path and Tree Systems in Graphs

A *separating system* $\mathcal{S}$ of a set X is a collection of subsets of X such that for any pair of distinct elements in X, there exists a set in $\mathcal{S}$ that contains exactly one of the two elements. In general, the main challenge is to find the size of the smallest separating system.

This problem lies at the intersection of combinatorics and optimization, making it a rich area of study with connections to coding theory, information theory, and graph theory [35, 37, 47]. For example, in the context of the identifying code problem, one seeks to separate the vertices in a graph G using a separating system defined by the closed neighbourhoods of vertices in $V(G)$ [47]. Determining the size of the smallest separating system becomes particularly challenging when additional structural constraints are imposed on the elements of the separating system.

We explore the concept of separating systems of vertex sets of graphs using paths and trees. Let G be a graph. A separating system of $V(G)$ is called a vertex-separating path (tree) system of G if the elements of the separating system are paths (trees) in G. In chapter 2, we focus on the size of the smallest vertex-separating path (tree) system for different types of graphs, including trees, grids, and maximal outerplanar graphs. Our results include improved lower bounds for the size of vertex-separating path systems of trees, a tight upper bound for the size of vertex-separating path systems of outerplanar graphs, and an upper bound for the size of vertex-separating tree systems of a graph.

This chapter of the thesis is a joint work with Ahmad Biniaz, Prosenjit Bose, Jean-Lou De Carufel, Anil Maheshwari, Babak Miraftab, Michiel Smid, Shakhar Smorodinsky, and Yelena Yuditsky and published in the journal of *Discrete Mathematics & Theoretical Computer Science* [12].

Chapter 3: Euclidean Freeze-Tag Problem on Plane

The freeze-tag problem is an optimization problem introduced by Arkin et al. (SODA'02). This problem revolves around efficiently waking up a swarm of inactive robots starting with a single active robot. Consider a set of $n+1$ robots for $n \in \mathbb{N}$. An active robot must travel to the position of an inactive robot to awaken (activate) it. As soon as an inactive robot is activated, it can assist in waking up the other inactive robots. We assume that each active robot moves at the same speed of one unit per second while the inactive robots do not move. The *makespan* (*wake-up time*) is the total time spent until the last robot is awakened. The objective, in the freeze tag problem, is to minimize the makespan. The freeze-tag problem has applications in group formation, searching, and recruitment in robotics, as well as broadcasting and IP multicast problems in network design (see [3, 50] and their references).

A recent paper by Bonichon et al. [16] considers the geometric version of the freeze-tag problem on the plane. It is conjectured that for robots located on the plane with ℓ_2-norm, the wake-up time is at most $(1+2\sqrt{2})r \approx 3.83r$, where r is the maximum distance between the initial active robot and any inactive robot. In chapter 3, we prove the conjecture for robots in convex position. Moreover, we show an upper bound of $4.63r$ for the wake-up time of robots in a disk of radius r in the ℓ_2-norm, improving the best known bound of $5\sqrt{2}r \approx 7.07r$.

This chapter of the thesis is joint work with Nicolas Bonichon, Cyril Gavoille, and Nicolas Hanusse and appeared in the proceedings of the *36th Canadian Conference on Computational Geometry (CCCG 2024)* [17].

Chapter 4: On Rainbow Turán Triangles in Planar Graphs

Mantel's theorem is one of the first classical theorems in extremal graph theory. This theorem states that if G is a graph on n vertices with $|E(G)| > \frac{n^2}{4}$, then G contains a triangle. Recently, Aharoni et al. [1] considered the multicolour version of Mantel's theorem. They proved that if G_1, G_2, G_3 are three graphs on a common set of n vertices V such that $|E(G_i)| > \frac{26-2\sqrt{7}}{81}n^2 (\approx 0.257n^2)$ for $1 \leq i \leq 3$, then there are three distinct vertices $w, u, v \in V$ so that $\{w, u\} \in E(G_1)$, $\{u, v\} \in E(G_2)$ and $\{v, w\} \in E(G_3)$. The triple $\{w, u, v\}$ is called a rainbow triangle in G_1, G_2, G_3. We ask the planar version of this question: Let $G_1, G_2, G_3, \cdots, G_k$ be a sequence of edge-maximal planar graphs on the same set of n vertices. What is the smallest value of k, as a function of n, such that we are guaranteed to have a rainbow triangle in the sequence $G_1, G_2, \ldots, G_k$?

In chapter 4, we establish a simple linear upper bound and we construct almost linear number of maximal planar graphs on the same set of n vertices without creating a rainbow triangle. This chapter of the thesis is a joint work with Prosenjit Bose, Vida Dujmović, and Pat Morin, and being prepared for publication.

Chapter 1

An Optimal Algorithm for Product Structure in Planar Graphs

Prosenjit Bose [1] Pat Morin[1] Saeed Odak[2]

Abstract. The *Product Structure Theorem* for planar graphs (Dujmović et al. *JACM*, **67**(4):22) states that any planar graph is contained in the strong product of a planar 3-tree, a path, and a 3-cycle. We give a simple linear-time algorithm for finding this decomposition as well as several related decompositions. This improves on the previous $O(n \log n)$ time algorithm (Morin. *Algorithmica*, **85**(5):1544–1558).

1.1 Introduction

For two graphs G and X, the notation $G \subseteq X$ denotes that G is isomorphic to some subgraph of X. The following *planar product structure theorems* have recently been used as a key tool in resolving a number of longstanding open problems on planar graphs, including queue number [27], nonrepetitive chromatic number [30], adjacency labelling [29], universal graphs [32], p-centered colouring [25], and vertex ranking [18].[3][4]

Theorem 1.1 (Dujmović et al. [27], Ueckerdt, Wood, and Yi [65])**.** *For any planar graph G, there exists:*

[1]School of Computer Science, Carleton University

[2]School of Electrical Engineering and Computer Science, University of Ottawa

[3]For graphs G and X, an *X-decomposition* of G is a collection $\mathcal{X} := (B_x : x \in V(X))$ of subsets of $V(G)$ called *bags* indexed by the vertices of X and such that (i) for each $v \in V(G)$, $X[\{x \in V(X) : v \in B_x\}]$ is connected; and (ii) for each $vw \in E(G)$, there exists some $x \in V(X)$ such that $\{v,w\} \subseteq B_x$. The *width* of $\mathcal{X}$ is $\max\{|B_x| : x \in V(X)\} - 1$. In the special case where X is a tree, $\mathcal{X}$ is called a *tree decomposition* of G. The *treewidth* $tw(G)$ of G is the minimum width of any tree decomposition of G.

[4]For two graphs G_1 and G_2, the *strong graph product* of G_1 and G_2, denoted $G_1 \boxtimes G_2$, is a graph whose vertex set is $V(G_1 \boxtimes G_2) := V(G_1) \times V(G_2)$ and that contains an edge between distinct vertices $v = (v_1, v_2)$ and $w = (w_1, w_2)$ if and only if (i) $v_1 = w_1$ and $v_2w_2 \in E(G_2)$; (ii) $v_2 = w_2$ and $v_1w_1 \in E(G_1)$; or (iii) $v_1w_1 \in E(G_1)$ and $v_2w_2 \in E(G_2)$.

(a) a planar graph H of treewidth at most 3 and a path P such that $G \subseteq H \boxtimes P \boxtimes K_3$ [27];
(b) a planar graph H of treewidth at most 4 and a path P such that $G \subseteq H \boxtimes P \boxtimes K_2$; and
(c) a planar graph H of treewidth at most 6 and a path P such that $G \subseteq H \boxtimes P$ [65].

In each of the applications of Theorem 1.1, the proofs are constructive and lead to algorithms whose running-time is dominated by the time required to compute the relevant decomposition. The proofs of each part of Theorem 1.1 are constructive and lead to $O(n^2)$ time algorithms as observed already by Dujmović et al. [27]. Morin [54] later showed that there exists an $O(n \log n)$ time algorithm to find the decomposition in Theorem 1.1.a. In the current chapter, we show that there exists a linear time algorithm for finding each of the three decompositions guaranteed by Theorem 1.1. This immediately gives an $O(n)$-time algorithm for each of the following problems on any n-vertex planar graph G:

- computing an $O(1)$-queue layout of G [27];
- nonrepetitively vertex-colouring G with $O(1)$ colours [30];
- assigning $(1 + o(1)) \log n$-bit labels to the vertices of G so that one can determine from the labels of vertices v and w whether or not v and w are adjacent in G [29];
- mapping the vertices of G into a universal graph U_n that has $n^{1+o(1)}$ vertices and edges so that any pair of vertices that are adjacent in G maps to a pair of vertices that are adjacent in U_n [32];
- colouring the vertices of G with $O(p^3 \log p)$ colours so that each connected subgraph H of G contains a vertex whose colour is unique in H or contains vertices of at least $p + 1$ different colours [25]; and
- colouring the vertices of G with $O(\log n / \log\log\log n)$ integers so that the maximum colour that appears on any path P of length at most ℓ appears at exactly one vertex of P (for any fixed $\ell \geq 2$) [18].

In addition to planar graphs, product structure theorems similar to Theorem 1.1 exist for k-planar graphs, h-framed graphs, bounded-genus graphs, and graphs from apex-minor-free families [9, 27, 28]. The existence of each of these decompositions relies at some point on Theorem 1.1. Thus, the algorithm presented here can also serve as an optimal subroutine in the computation of product structure decompositions of graphs from these classes. However, these larger graph classes also require additional machinery that (at the time of writing) makes finding these decompositions largely impractical.[5]

The remainder of this chapter is organized as follows: Section 1.2 presents some necessary background and notation. Section 1.3 reviews the proof of Theorem 1.1.a. Section 1.4 presents the linear time algorithm for finding the decomposition in Theorem 1.1.a. Section 1.5 describes the algorithms for finding the decompositions in Theorem 1.1.b and Theorem 1.1.c.

[5]Possible exceptions here are cases in which a bounded-genus graph, k-planar graph, or h-framed graph is given along with its embedding.

1.2 Preliminaries

Throughout this chapter we use standard graph theory terminology as used in the textbook by Diestel [26]. All graphs discussed here are simple and finite. For a graph G, $V(G)$ and $E(G)$ denote the vertex and edge sets of G, respectively. We use the terms *vertex* and *node* interchangeably, though we typically refer to the vertices of some primary graph G of interest and refer to the nodes of some auxilliary graph (such as a spanning tree) related to G. We say that a subgraph G' of a graph G *spans* a set $S \subseteq V(G)$ if $S \subseteq V(G')$.

Quotient Graphs. Given a graph G and a partition $\mathcal{P}$ of $V(G)$, the *quotient graph* $G/\mathcal{P}$ is the graph with vertex set $V(G/\mathcal{P}) := \mathcal{P}$ and in which two nodes $X, Y \in V(G/\mathcal{P})$ are adjacent if G contains at least one edge xy with $x \in X$ and $y \in Y$.

Embeddings, Planar Graphs, and (Near-)Triangulations. An *embedding* ψ of a graph G associates each vertex v of G with a point $\psi(v) \in \mathbb{R}^2$ and each edge vw of G with a simple open curve $\psi(vw) : (0,1) \to \mathbb{R}^2$ whose endpoints[6] are $\psi(v)$ and $\psi(w)$. We do not distinguish between such a curve $\psi(vw)$ and the point set $\{\psi(vw)(t) : 0 < t < 1\}$. We let $\psi(V(G)) := \{\psi(v) : v \in V(G)\}$, $\psi(E(G)) := \bigcup_{vw \in E(G)} \psi(vw)$, and $\psi(G) := \psi(V(G)) \cup \psi(E(G))$. An embedding ψ of G is *plane* if $\psi(vw) \cap \psi(V(G)) = \emptyset$ and $\psi(vw) \cap \psi(xy) = \emptyset$ for each distinct pair of edges $vw, xy \in E(G)$. A graph G is *planar* if it has a plane embedding. A *triangulation* is an edge-maximal planar graph.

If ψ is a plane embedding of a planar graph G, then we call the pair (G, ψ) an *embedded graph* and we will not distinguish between a vertex v of G and the point $\psi(v)$ or between an edge vw of G and the curve $\psi(vw)$. Similarly, we will not distinguish between G and the point set $\psi(G)$. Any cycle in an embedded graph defines a Jordan curve. For such a cycle C, $\mathbb{R}^2 \setminus C$ has two components, one bounded and the other unbounded. We will refer to the bounded component as the *interior* of C and the unbounded component as the *exterior* of C. If G is an embedded triangulation, then the subgraph of G consisting of all edges and vertices of G contained in the closure of the interior of C is called a *near-triangulation*.

Each component of $\mathbb{R}^2 \setminus G$ is a *face* of G and we let $F(G)$ denote the set of faces of G. If G is 2-connected then, for any face $f \in F(G)$, the set of vertices and edges of G contained in the boundary of f forms a cycle. We may therefore treat a face f of a 2-connected graph G as a component of $\mathbb{R}^2 \setminus G$ or as the cycle of G on the boundary of f, relying on context to distinguish between the two usages. Note that every embedded graph contains exactly one face—the *outer face*—that is unbounded.

Duals and Cotrees. The *dual* $G^\star$ of an embedded graph G is the graph with vertex set $V(G^\star) := F(G)$ and edge set $E(G^\star) := \{fg \in \binom{F(G)}{2} : E(f) \cap E(g) \neq \emptyset\}$.[7] If T is a spanning tree of G then the *cotree* $\overline{T}$ of (G, T) is the graph with vertex set $V(\overline{T}) := V(G^\star)$ and

[6] The *endpoints* of an open curve $\psi : (0,1) \to \mathbb{R}^2$ are the two points $\lim_{\epsilon \downarrow 0} \psi(\epsilon)$ and $\lim_{\epsilon \downarrow 0} \psi(1-\epsilon)$.

[7] For a set S, $\binom{S}{2}$ denotes the $\binom{|S|}{2}$-element set $\binom{S}{2} := \{\{x,y\} : x, y \in S, x \neq y\}$.

edge set $E(\overline{T}) := \{ab \in E(G^\star) : E(a) \cap E(b) \setminus E(T) \neq \emptyset\}$. It is well known that, if G is connected and T is a spanning tree of G then $\overline{T}$ is a spanning tree of $G^\star$ [26, Chapter 4, Exercise 42].

For our purposes, a *binary tree* is a rooted tree of maximum degree 3 whose root has degree at most 2 and in which each child v of a node u is either the unique *left child* or the unique *right child* of u. If G is a triangulation and we root $\overline{T}$ at any face $f_0 \in F(G)$ that contains an edge of T, then $\overline{T}$ is a binary tree, with the classification of left and right children determined by the embedding of G.[8]

Paths and Distances. A *path* in G is a (possibly empty) sequence of distinct vertices $v_0,\dots,v_r$ with the property that $v_{i-1}v_i \in E(G)$, for each $i \in \{1,\dots,r\}$. The *endpoints* of a path $v_0,\dots,v_r$ are the vertices v_0 and v_r. The *length* of a non-empty path $v_0,\dots,v_r$ is the number, r, of edges in the path.

Trees, Depth, Ancestors, and Descendants. Let T be a tree rooted at a vertex $v_0 \in V(T)$. For any vertex $w \in V(T)$, $P_T(w)$ denotes the path in T from w to v_0. For any $w_0 \in V(T)$, any prefix $w_0,\dots,w_r$ of $P_T(w_0)$ is called an *upward path* in T; w_0 is the *lower endpoint* of this path and w_r is the *upper endpoint*. The *T-depth* of a node $w \in V(T)$ is the length of the path $P_T(w)$. The second node in $P_T(v)$ (if any) is the *T-parent* of v. A vertex $a \in V(T)$ is a *T-ancestor* of $w \in V(T)$ if $a \in V(P_T(w))$. If a is a T-ancestor of w then w is a *T-descendant* of a.

Lowest Common Ancestors. For any two vertices $v,w \in V(T)$, the *lowest common ancestor* $\mathrm{lca}_T(v,w)$ of v and w is the node a in $P_T(v) \cap P_T(w)$ having maximum T-depth. The *lowest commmon ancestor problem* is a well-studied data structuring problem that asks to preprocess a given n-vertex rooted tree so that one can quickly return $\mathrm{lca}_T(v,w)$ for any two nodes $v,w \in V(T)$. A number of optimal solutions to this problem exist that, after $O(n)$ time preprocessing using $O(n)$ space, can answer queries in $O(1)$ time [2, 10, 11, 34, 42, 61]. The most recent work in this area includes simple and practical data structures that achieve this optimal performance [2, 10, 34].

Reconstructing Binary Tree Models. Let T be a binary tree and $S \subseteq V(T)$. An upward path $v_0,\dots,v_r$ in a binary tree T is *S-non-branching* if v_i has degree 2 and $v_i \notin S$ for each $i \in \{1,\dots,r-1\}$. For any binary tree T and set $S \subseteq V(T)$, the *model* T' of T with respect to S is the binary tree obtained by replacing each maximal S-non-branching path $v_0,\dots,v_r$ with the edge v_0v_r; if v_{r-1} is the left (respectively, right) child of v_r then v_0 becomes the left (respectively, right) child of v_r. See Fig. 1.1.

Lemma 1.1. *Let T be a binary tree, let $S = \{x_1,\dots,x_d\} \subseteq V(T)$, and let T_0 be the minimal subtree of T that spans S. Then there exists an algorithm that, given an $O(1)$-query time lowest common ancestor data structure for T, computes the model T_0' of T_0 with respect to S in $O(d^2)$ time.*

[8] There is a small ambiguity here when T contains two edges of f_0, in which case the unique child of f_0 in $\overline{T}$ can be treated as the left or right child of f_0.

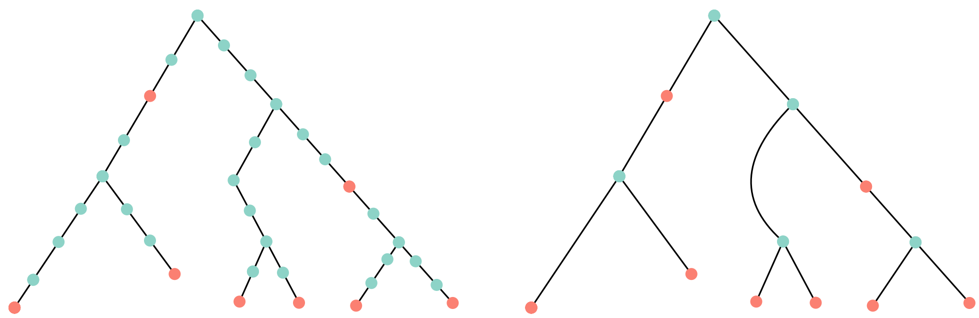


Fig. 1.1: A binary tree T with set $S \subseteq V(T)$ depicted in red and the model of T with respect to S.

Proof. The proof is by induction on $|S|$. The base case $|S| = 1$ is trivial, since then $T_0' = T_0$ is the tree with one node, which is the unique element in S.

If $|S| \geq 2$, then the first step is to determine the root r of T_0, which must also be the root of T_0'. This is easily done by first setting $r := x_1$ and then repeatedly setting $r := \mathrm{lca}_T(r, x_i)$ for each $i \in \{2, \dots, d\}$. This step takes $O(d)$ time.

If r has no left child in T, then we can immediately apply induction on $S \setminus \{r\}$ and make the right child of r in T_0' the root of the model obtained by induction. The case in which r has no right child can be handled similarly. If r has both a left child r_1 and a right child r_2, then the next step is to partition $S \setminus \{r\}$ into a set S_1 of descendants of r_1 and a set S_2 of descendants of r_2. For each $x \in S \setminus \{r\}$ there are only two possibilities for $\mathrm{lca}_T(r_1, x)$

1. If $\mathrm{lca}_T(r_1, x) = r_1$ then $x \in S_1$.
2. If $\mathrm{lca}_T(r_1, x) = r$ then $x \in S_2$.

Therefore, using $O(d)$ lowest common ancestor queries, we can determine the root r of T' and partition $S \setminus \{r\}$ into sets S_1 and S_2 that define the left and right subtrees of r. We can now recurse on S_1 to obtain a tree with root r_1' and recurse on S_2 to obtain a tree with root r_2'. We make r_1' the left child of r and r_2' the right child of r to obtain the model T_0' of T_0. The running-time of this algorithm obeys the recurrence $T(d) \leq O(d) + T(d_1) + T(d_2)$, where $d_1 + d_2 \leq d$ and $d_1, d_2 \leq d - 1$. This recurrence resolves to $T(d) \in O(d^2)$. □

1.3 Tripod Decompositions

Refer to Fig. 1.2. Let G be an n-vertex triangulation and let T be a spanning tree of G. For a face uvw of G, a (G,T)-*tripod* Y with *crotch* uvw is the vertex set of three disjoint (and each possibly empty) upward paths in T (the *legs* of Y) whose lower endpoints are u, v, and w. A (G,T)-*tripod decomposition* is a partition of $V(G)$ into (G,T)-tripods. Dujmović et al. [27] proved the following result:

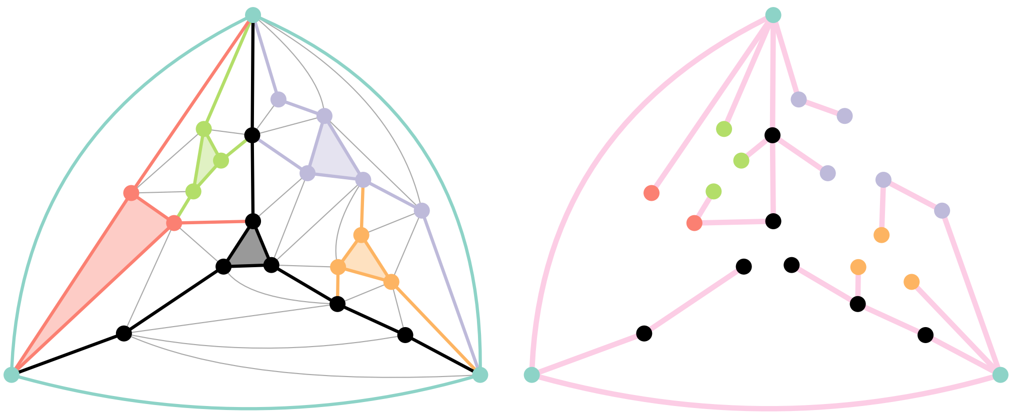


Fig. 1.2: A (G,T)-tripod decomposition of a triangulation G (and the underlying spanning tree T.)

Theorem 1.2. *Let G be a triangulation and T be a spanning tree of G. Then there exists a (G,T)-tripod decomposition $\mathcal{Y}$ such that $G/\mathcal{Y}$ has treewidth at most* 3.

It is straightforward to verify that Theorem 1.2 implies Theorem 1.1.a by first triangulating the given graph and then taking T to be a breadth-first spanning tree of the resulting triangulated graph [27, Observation 35].

1.3.1 Tripod Decompositions from Face Orderings

We now describe how a (G,T)-tripod decompositions can be obtained from a sequence of distinct faces of G. Throughout this section (and for the remainder of the chapter):

- G is an embedded triangulation with outer face f_0 and
- T is a spanning-tree of G rooted at a vertex $v_0 \in V(f_0)$.

For any subgraph f of G, we define $Y_T(f) := f \cup \bigcup_{v\in V(f)} P_T(v)$.[9] In words, $Y_T(f)$ is the subgraph of G that includes all the vertices and edges of f and all the vertices and edges of each path from each vertex of f to the root of T.

Let $\mathcal{F} := f_0,\dots,f_r$ be a sequence of distinct faces of G whose first element is the outer face f_0. Let G_{-1} denote the graph with no vertices and, for each $i \in \{0,\dots,r\}$, define the graph $G_i := \bigcup_{j=0}^{i} Y_T(f_j)$ and let $Y_i := V(G_i)\setminus V(G_{i-1})$. Let $\mathcal{G}_{\mathcal{F}} := G_0,\dots,G_r$ and let $\mathcal{Y}_{\mathcal{F}} := Y_0,\dots,Y_r$.

Informally, we require that each of the *legs* of each tripod Y_i have a *foot* on a different vertex of G_{i-1} and that the tripods $Y_1,\dots,Y_r$ cover all the vertices and edges of G. Formally, we say that the sequence $\mathcal{F}$ is *proper* if, for each $i \in \{1,\dots,r\}$, and each distinct $v,w \in V(f_i)$, $V(Y_T(v)\cap G_{i-1}) \neq V(Y_T(w)\cap G_{i-1})$. The sequence $\mathcal{F}$ is *complete* for G if $G_r = G$. Note that, if $\mathcal{F}$ is complete, then $\{Y_0,\dots,Y_r\}$ is a tripod decomposition of G.

[9] In all of our examples, the subgraph f will always be a single edge or single face of G.

From the preceding definitions it follows that, if $\mathcal{F}$ is proper, then G_i is 2-connected for each $i \in \{0,\ldots,r\}$. For any $i \in \{0,\ldots,r\}$, consider any face f of G_i, that we now treat as a cycle in G. An easy proof by induction shows that, for any $j \in \{0,\ldots,i\}$, the induced graph $f[Y_j]$ is connected. We are interested in keeping the number of tripods in $Y_0,\ldots,Y_i$ that contribute to $V(f)$ as small as possible, which motivates our next definition.

The sequence $\mathcal{F}$ is *good* if the resulting sequence of graphs $\mathcal{G}_{\mathcal{F}} := G_0,\ldots,G_r$ and tripods $\mathcal{Y}_{\mathcal{F}} := Y_0,\ldots,Y_r$ satisfy the following condition: For each $i \in \{0,\ldots,r\}$ and each face f of G_i,

$$|\{\ell \in \{0,\ldots,i\} : V(f) \cap Y_\ell \neq \emptyset\}| \leq 3 \ .$$

In words, each face of each graph G_i has vertices from at most three tripods of $Y_0,\ldots,Y_i$ on its boundary. Even more, the vertices of f can be partitioned into at most three paths where the vertices of each path belong to a single tripod. Dujmović et al. [27] prove Theorem 1.2 by proving the next lemma.

Lemma 1.2. *Let G be a triangulation with a vertex v_0 on its outer face f_0 and let T be a spanning tree of G rooted at v_0. Then there exists a sequence $\mathcal{F} := f_0,\ldots,f_r$ of distinct faces of G that is proper, good, and complete.*

Remark 1.1. *Lemma 1.2 is stated in terms of sequences only for convenience and could be rephrased in terms of partial orders. Indeed, consider the partial order $\prec$ defined as follows: For each $i \in \{1,\ldots,r\}$ let f_i' be the face of G_{i-1} that contains f_i; then $f_\ell \prec f_i$ for each $\ell \in \{0,\ldots,i-1\}$ such that $V(f_i') \cap Y_\ell \neq \emptyset$. It is straightforward to check that any linearization of this partial order will result in the same tripod decomposition $\mathcal{Y}_{\mathcal{F}} := \{Y_0,\ldots,Y_r\}$.*

Dujmović et al. [27] prove Lemma 1.2 by giving a recursive algorithm that constructs the face sequence $\mathcal{F}$. For a face f of G_i, define the set $I_f := \{\ell \in \{0,\ldots,i\} : V(f) \cap Y_\ell \neq \emptyset\}$. They begin with the outer face f_0 of G. To find the face f_i, $i > 0$, they consider some face $f \notin \{f_0,\ldots,f_{i-1}\}$ of G_{i-1} and use Sperner's Lemma to show that there is an appropriate face f_i of G (called a *Sperner triangle*) that is contained in f. In particular, f_i is chosen so that the three upward paths in $Y_F(f_i)$ lead back to each of the (at most 3) tripods in $\{Y_j : j \in I_f\}$. See Fig. 1.3.

This proof leads to a divide-and-conquer algorithm: After finding f_i, the algorithm recursively decomposes each of the near-triangulations that are bounded by the at most three new faces in $S_i := F(G_i) \setminus F(G_{i-1}) \setminus \{f_i\}$. The Sperner triangle f_i can easily be found in time proportional to the number of faces of G in the interior of f. However, because the resulting recursion is not necessarily balanced, a straightforward implementation of this yields an algorithm with $\Theta(n^2)$ worst-case running time.

Morin [54] later showed that, using an appropriate data structure for T, this approach can be implemented in such a way that the resulting algorithm runs in $O(n \log n)$ time. Essentially, Morin's algorithm works by finding the Sperner triangle f_i in time proportional to the minimum number of faces of G contained in any of the faces in S_i. In the next section, we will show that, by using a lowest common ancestor data

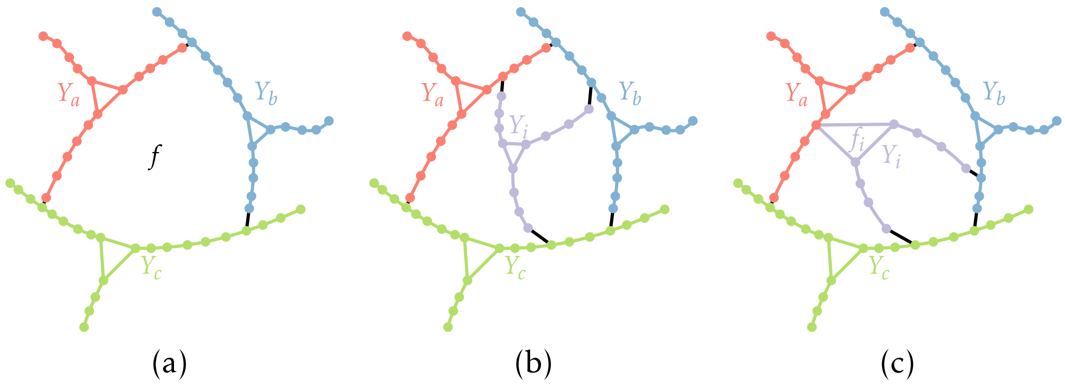


Fig. 1.3: Each face f in G_{i-1} is bounded by at most three tripods Y_{a_f}, Y_{b_f}, and Y_{c_f} and the tripod Y_i is chosen so that it connects each of these.

structure for the cotree $\overline{T}$ along with Lemma 1.1, the Sperner triangle f_i can be found in constant time, yielding an $O(n)$ time algorithm.

By now, our presentation of this material differs somewhat from that in [27, 65]. Therefore, we now pause to explain how Lemma 1.2 implies Theorem 1.2. Let G be a triangulation, let T be spanning tree of G, let $\mathcal{F} := f_0, \dots, f_r$ be the proper good face sequence guaranteed by Lemma 1.2, and let $\mathcal{Y}_\mathcal{F} := \{Y_0, \dots, Y_r\}$ be the resulting tripod decomposition. We now show that there exists a chordal graph H whose largest clique has size at most 4 and that contains $G/\mathcal{Y}_\mathcal{F}$. We construct the graph H so that for each $i \in \{0, \dots, r\}$ and each face f of G_i, H contains a clique on $\{Y_j : j \in I_f\}$. To accomplish this, for each $i \in \{1, \dots, r\}$ let f be the face of G_{i-1} that contains f_i and form a clique on $\{Y_i\} \cup \{Y_j : j \in I_f\}$. Inductively, the elements of $\{Y_j : j \in I_f\}$ already form a clique, so this operation is equivalent to attaching Y_i to all the vertices of an existing clique of size at most 3. Therefore, this results in a chordal graph H whose largest clique has size at most 4 and therefore H has treewidth at most 3 [38].

1.4 An $O(n)$-Time Algorithm

Refer to Fig. 1.4 for an illustration of the following (probably well-known) simple lemma, which is closely related to Sperner's Lemma:

Lemma 1.3. *Let N be a near-triangulation with outer face $v_0, \dots, v_r$ and colour each vertex of N red or blue in such a way that $v_0, \dots, v_\ell$ are coloured red for some $\ell \in \{0, \dots, r-1\}$ and $v_{\ell+1}, \dots, v_r$ are coloured blue. Then there exists a path $w_0, \dots, w_k$ in $N^\star$ such that*

1. *w_0 is the inner face of N with $v_0 v_r$ on its boundary;*
2. *w_k is the inner face of N with $v_\ell v_{\ell+1}$ on its boundary; and*
3. *for each $i \in \{1, \dots, k\}$, the single edge in $E(w_{i-1}) \cap E(w_i)$ has an endpoint of each colour.*

Proof. If $w_0 = w_k$, the lemma is immediately true, so assume $w_0 \neq w_k$. Say that an edge of N is *bichromatic* if one of its endpoints is red and the other is blue. Any edge that is

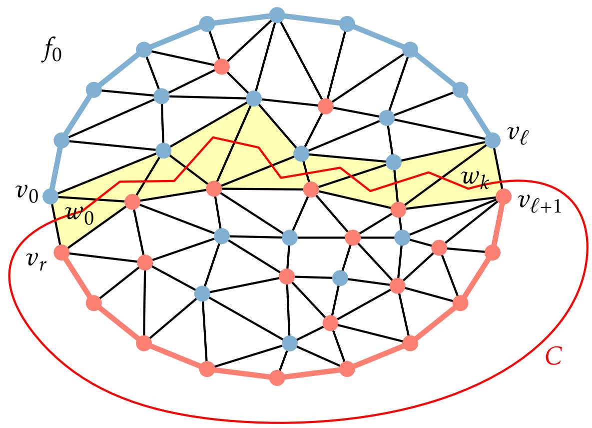


Fig. 1.4: Lemma 1.3

not bichromatic is *monochromatic*. The outer face f_0 of N has exactly two bichromatic edges v_0v_r and $v_\ell v_{\ell+1}$ and any inner face of N has either zero or two bichromatic edges. Consider the subgraph H of $N^\star$ obtained removing each edge $fg \in E(N^\star)$ such that the edge in $E(f) \cap E(g)$ is monochromatic. Every vertex in H has degree 0 or 2, so each connected component of H is either an isolated vertex or a cycle. The face f_0 has degree 2 so it is contained in a cycle C of H. The two neighbours of f_0 in H are w_0 and w_k. Therefore C contains a path $w_0, \dots, w_k$ that satisfies the conditions of the lemma. □

The next lemma, which is the main new insight in this chapter, allows us to use Lemma 1.1 to find Sperner triangles in constant time.

Lemma 1.4. *Let G be a triangulation with a vertex v_0 on its outer face f_0; let T be a spanning tree of G rooted at v_0; let $\overline{T}$ be the cotree of (G,T) rooted at f_0; let $f_0, \dots, f_{i-1}$ be a good proper sequence of faces of G that yields a sequence $\mathcal{G}_\mathcal{F} := G_0, \dots, G_{i-1}$ of graphs and a sequence $\mathcal{Y}_\mathcal{F} := Y_0, \dots, Y_{i-1}$ of tripods; let $f \notin \{f_0, \dots, f_{i-1}\}$ be a face of G_{i-1}, and let $S \subseteq F(G)$ contain exactly the (at most three) faces $g \in F(G)$ such that*

(i) *g is contained in the interior of f;*
(ii) *g contains an edge $vw \in E(f)$ with $v \in Y_a$ and $w \in Y_b$ for some distinct $a, b \in I_f$.*

Let $\overline{T}_0$ be the minimal subtree of $\overline{T}$ that spans S. Then, if S is non-empty and $f_i \in V(\overline{T}_0)$ is such that each component of $\overline{T}_0 - f_i$ contains at most one element of S, Then $f_0, \dots, f_i$ is good.

Proof. Let N be the near-triangulation consisting of all vertices and edges of G contained in the closure of the interior of f. Recall that $I_f := \{j \in \{0, \dots, i-1\} : Y_j \cap V(f) \neq \emptyset\}$. Since $f_0, \dots, f_{i-1}$ is good, $|I_f| \leq 3$. Since S is non-empty $|I_f| \geq 2$. For each $j \in I_f$, colour each vertex v of N with the colour j if the first vertex of $P_T(v)$ in $V(f)$ is contained in Y_j. Say that an edge or face of N is *monochromatic*, *bichromatic*, or *trichromatic* if it contains vertices of one, two, or three colours, respectively.

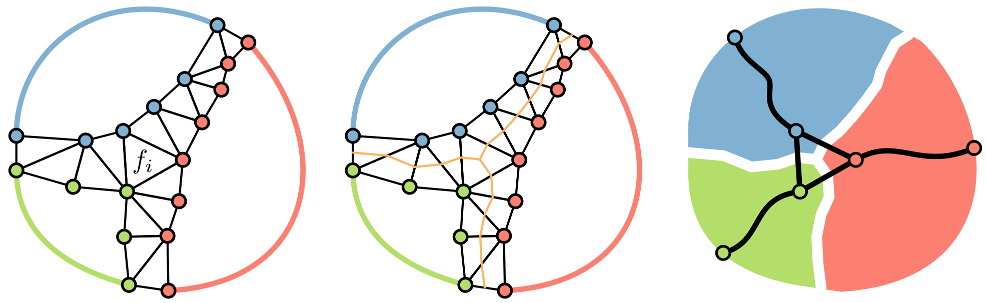


Fig. 1.5: The proof of Lemma 1.4

$E(f)$ contains exactly $|I_f|$ bichromatic edges. Since each element of S is an inner face of N that contains a bichromatic edge of f, $|S| \leq |I_f| \leq 3$. Let X be the subgraph of $N^\star$ that contains an edge $fg \in E(N^\star)$ if and only if f and g are inner faces of N and the edge in $E(f) \cap E(g)$ is bichromatic. We claim that X is a subgraph of $\overline{T}$. In order to show this, we need only argue that each edge uv of T in the interior of f is monochromatic. Consider any $uv \in E(N) \setminus E(f)$ where u is the T-parent of v. If $v \notin V(f)$ then, by definition, v has the same colour as u, so uv is monochromatic. The case where $v \in V(f)$ and $u \notin V(f)$ can not occur since $v \in V(f)$ implies that $P_T(v) \subseteq G_{i-1}$, but $u \notin V(G_{i-1})$. Similarly, the case in which $u \in V(f)$ and $v \in V(f)$ can not occur since this implies that $P_T(v) \subseteq G_{i-1}$, but $uv \notin E(G_{i-1})$.

Next we claim that all the elements of S are in a single connected component of X. If $|I_f| = 2$, then this follows immediately from Lemma 1.3. If $|I_f| = 3$, then let $\{a, b, c\} := I_f$ and consider a pair $g_1, g_2 \in S$ where (without loss of generality) g_1 contains a bichromatic edge of f with colours a and b and g_2 contains a bichromatic edge of f with colours b and c. By treating a and c as a single colour we may again apply Lemma 1.3 to conclude that g_1 and g_2 are in the same component of X.

Refer to Fig. 1.5(a). Therefore X is a subgraph of $\overline{T}$ that has a component containing all the elements of S. Therefore X contains $\overline{T}_0$. By choice, $\overline{T}_0$ contains a path from f_i to each $g \in S$ and each of these paths is disjoint except for their shared starting location f_i.

Refer to Fig. 1.5(b). Now, consider the embedded graph X_0 obtained as follows: For each $g \in V(\overline{T}_0)$, place a vertex on the center of each bichromatic edge of g and, if g is trichromatic, then place a vertex in the center of g. Next,

1. add an edge joining the center of each trichromatic triangle to each of the centers of its bichromatic edges; and
2. add an edge (embedded as a straight line segment) joining the centers of each pair of bichromatic edges that are on a common bichromatic face $g \in V(\overline{T}_0)$.

The graph X_0 is a tree of maximum-degree 3 that has $|I_f|$ leaves. (Each leaf in X_0 is the center of a bichromatic edge in $E(f)$). With the exception of these three leaves, every point in the embedding of X_0 is contained in the interior of f.

Refer to Fig. 1.5(c). Now treat X_0 as a point set and consider the point set f' obtained by removing X_0 from the closure of f. Now f' has $|I_f|$ connected components and each vertex of f_i is in a different component. Each of the components of f' contains vertices of Y_j for exactly one $j \in I_f$; call this the *colour* of the component. Since no edge of T crosses X_0, the colour of each vertex in f_i is equal to the colour the component of f' that contains it.

Finally, to see that $f_0, \ldots, f_i$ is good first observe that we need only be concerned with the at most three faces in $F(G_i) \setminus F(G_{i-1}) \setminus \{f_i\}$ and each of these shares a bichromatic edge with f_i. If g is a face in $F(G_i) \setminus F(G_{i-1}) \setminus \{f_i\}$ with $E(g) \cap E(f_i) = \{uv\}$ and uv is coloured with a and b, then $V(g) \cap Y_j = \emptyset$ for any $j \in \{0, \ldots, i\} \setminus \{a, b, i\}$. This completes the proof. □

Theorem 1.3. *There exists an $O(n)$ time algorithm that, given any n-vertex triangulation G and any rooted spanning tree T of G, produces a (G,T)-tripod decomposition $\mathcal{Y}$ such that* $\mathrm{tw}(G/\mathcal{Y}) \leq 3$.

Proof. Let v_0 be the root of T and let f_0 be a face of G incident to v_0 that contains an edge of T incident to v_0. In a preprocessing step, we compute the cotree $\overline{T}$ of (G,T) and construct a lowest common ancestor data structure for $\overline{T}$ in $O(n)$ time that allows us to compute $\mathrm{lca}_{\overline{T}}(f,g)$ for any two faces $f, g \in F(G)$ in $O(1)$ time.

After this preprocessing, we construct the good sequence $f_0, \ldots, f_r$ recursively. Conceptually, during any recursive invocation, the input is a near-triangulation N bounded by a cycle C in G whose vertices belong to at most three tripods computed in previous steps. Each vertex of G starts initially *unmarked* and we *mark* a vertex once we have placed it in a tripod. The precise input to a recursive invocation is defined as follows:

1. If C intersects three tripods then the input consists of the three inner faces g_1, g_2, and g_3 of N that contain bichromatic edges of C. Lemma 1.4 characterizes the face f_i in terms of the minimum subtree $\overline{T}_0$ of $\overline{T}$ that contains g_1, g_2, and g_3. Indeed, f_i is either the unique degree-3 node of $\overline{T}_0$ (if g_1, g_2, and g_3 are all leaves of $\overline{T}_0$) or f_i is the unique node among g_1 g_2, or g_3 that has degree 2. By Lemma 1.1 we can construct the model $\overline{T}_0'$ of $\overline{T}_0$ in constant time and find the node f_i.

2. If C intersects two tripods, then the input consists of two inner faces g_1, g_2, of N with bichromatic edges of C on their boundary. In this case, we let $f_i := g_1$ or $f_i = g_2$, either choice satisfies our requirements.

3. If C intersects only one tripod, then the input consists of any inner face g_1 of N that contains an edge in $E(f)$. In this case $f_i := g_1$ satisfies our requirements.

Once we have found the Sperner triangle f_i, we can compute the tripod Y_i and mark its vertices by following the path in T from each vertex of f_i to its nearest marked ancestor in T. This takes $O(1 + |Y_i|)$ time. Once we have done this, we have also found the at most three bichromatic edges of G_i that are needed to perform the at most three recursive invocations on the near triangulations whose outer faces coincide with each of the new faces in $F(G_i) \setminus F(G_{i-1}) \setminus \{f_i\}$.

After setting f_0, the initial recursive call falls into the third case above, so its input is any of the three inner faces that shares an edge with the outer face, f_0. Each recursive invocation adds a new face f_i to the good face sequence $f_0, \ldots, f_r$ and takes $O(1+|Y_i|)$ time. Since $Y_0, \ldots, Y_r$ is a partition of $V(G)$, the running time of this algorithm is therefore $\sum_{i=0}^{r} O(1+|Y_i|) = O(n)$. □

1.5 Variations

In this section we show that there are $O(n)$ time algorithms for computing the decompositions in Theorem 1.1.b and Theorem 1.1.c. In the same way that Theorem 1.1.a follows from the tripod decomposition of Theorem 1.2, Theorem 1.1.b follows from a bipod decomposition given by Theorem 1.4 and Theorem 1.1.c follows from a monopod decomposition given by Theorem 1.5.

1.5.1 Bipod Decompositions

We begin with the decomposition in Theorem 1.1.b, which was communicated to us by Vida Dujmović, and has not appeared before. This decomposition is obtained by selecting a proper sequence $\mathcal{E} := e_0, \ldots, e_k$ of distinct edges of G, which define a sequence of graphs $\mathcal{G}_{\mathcal{E}} := G_0, \ldots, G_k$ where $G_i := \bigcup_{j=0}^{i} P_T(e_j)$ and a sequence of *bipods* $\mathcal{I}_{\mathcal{E}} := \Lambda_0, \ldots, \Lambda_k$ where $\Lambda_i = V(G_i) \setminus V(G_{i-1})$. We call $\mathcal{E}$ *good* if, for each $i \in \{0, \ldots, k\}$ and each face $f \in F(G_i)$, $V(f)$ has a non-empty intersection with at most 4 bipods in $\Lambda_0, \ldots, \Lambda_i$.

Exactly the same argument used in Section 1.3.1 to show that $G/\mathcal{Y}_{\mathcal{F}}$ is contained in a chordal graph of maximum clique size 4 also shows that if $\mathcal{E}$ is a good edge sequence that produces a bipod partition $\mathcal{I}_{\mathcal{E}}$ of $V(G)$, then $G/\mathcal{I}_{\mathcal{E}}$ is contained in a chordal graph of maximum clique size 5, so $G/\mathcal{I}_{\mathcal{E}}$ has treewidth at most 4.

We now explain why a good edge sequence $e_0, \ldots, e_r$ exists.[10] As before, we set f_0 to be any face of G such that $E(f_0)$ contains an edge of T incident to the root v_0 of T. The edge e_0 is any edge of $E(f_0) \setminus E(T)$. Next we take special care to ensure that G_i is biconnected for $i \geq 1$. In particular, if G_0 contains only two edges of f_0, then we take e_1 to be the edge of f_0 that does not appear in G_0. Otherwise, we choose e_1 using the general strategy for choosing e_i, described next.

Refer to Fig. 1.6. Now we may assume that G_{i-1} is biconnected. To choose the edge e_i, we consider any face $f \in F(G_{i-1}) \setminus F(G)$. Inductively, $V(f)$ contains vertices from at most four bipods in $\Lambda_0, \ldots, \Lambda_{i-1}$. Let $I_f := \{j \in \{0, \ldots, i-1\} : \Lambda_j \cap V(f) \neq \emptyset\}$. If $|I_f| < 4$ then we can select e_i to be any edge in the interior f. Therefore, we focus on the case

[10]The existence of this edge sequence is more easily proven using Sperner's Lemma, but we want a proof that lends itself to a linear time algorithm.

$|I_f| = 4$. As before we colour vertices in the near triangulation N using colours in the set I_f; we let S be the set of inner faces in N that contain a bichromatic edge in $E(f)$; and let $\overline{T}_0$ be the minimal subtree of $\overline{T}$ that spans S. The same argument in the proof of Lemma 1.4 shows that every node of $\overline{T}_0$ is contained in f.

Claim 1.1, below, shows that $\overline{T}_0$ contains an edge xy such that each component of $\overline{T}_0 - xy$ contains at most two elements of S. It is straightforward to verify that, if we choose e_i to the be the edge in $E(x) \cap E(y)$ then we obtain a graph G_i in which each of the two new faces containing vertices from Λ_i contains vertices from at most three bipods in $\{\Lambda_j : j \in I_f\}$, as required.

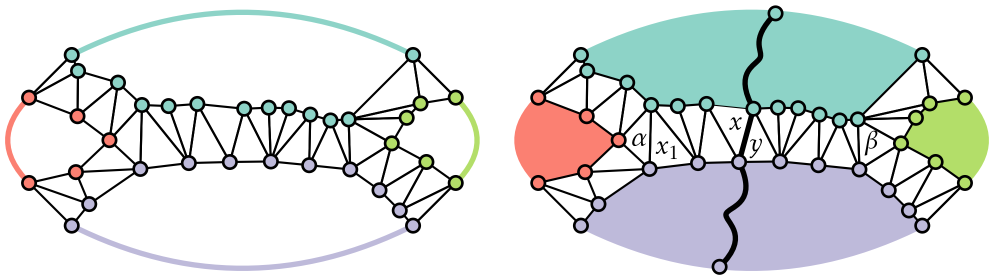


Fig. 1.6: Choosing the next e_i in a good edge sequence.

Claim 1.1. *$\overline{T}_0$ contains an edge xy such that each component of $\overline{T}_0 - xy$ contains at most two nodes of S.*

Proof. Direct each edge xy of $\overline{T}_0$ in the direction $\overrightarrow{xy}$ if the component of $\overline{T}_0 - xy$ that contains y contains three or more nodes of S. It is sufficient to show that this process leaves some edge xy of $\overline{T}_0$ undirected. Assume for the sake of contradiction that every edge of $\overline{T}_0$ is directed. Then some node x of $\overline{T}_0$ has only incoming edges. Certainly x does not have degree 1 in $\overline{T}_0$.

If x has degree 2 in $\overline{T}_0$ then $\overline{T}_0$ contains two subtrees T_1 and T_2 that have only the node x in common and such that $|V(T_1) \cap S| \geq 3$ and $|V(T_2) \cap S| \geq 3$, which implies that $|S| \geq 3 + 3 - 1 > 4$, a contradiction.

Suppose therefore that x has degree 3 in $\overline{T}_0$. Each face in S contains an edge in $E(f)$, so each face in S has degree at most 2 in $\overline{T}_0$. Therefore $x \notin S$. Therefore $\overline{T}_0 - x$ contains three components T_1, T_2, T_3 such that each pair of components contains at least 3 elements of S. But this implies that $|S| \geq (3 \times 3)/2 > 4$, a contradiction. □

Algorithmically, using Lemma 1.1, we can construct the model $\overline{T}_0'$ of $\overline{T}_0$ in constant time given the set S. The model $\overline{T}_0'$ will also contain an edge $\alpha\beta$ such that each component of $\overline{T}_0' - \alpha\beta$ contains at most two nodes in S. We claim that $E(\alpha)$ contains an edge that makes a suitable choice for e_i, and this edge can be found in constant time.

Indeed, the edge $\alpha\beta$ in $\overline{T}_0'$ corresponds to a path $\alpha, x_1, \ldots, x_k, \beta$ in $\overline{T}_0$ and the unique edge in $E(\alpha) \cap E(x_1)$ is a suitable choice for e_i.

The rest of the details of the algorithm are similar to those given in the proof of Theorem 1.3: Each subproblem is a near-triangulation N bounded by a cycle C and the input that defines the subproblem consists of the (at most four) faces $S \subseteq F(N)$ incident to bichromatic edges of C.[11]

Theorem 1.4. *There exists an $O(n)$ time algorithm that, given any n-vertex triangulation G and any rooted spanning tree T of G, produces a (G,T)-bipod decomposition $\mathcal{I}$ such that* $\operatorname{tw}(G/\mathcal{I}) \leq 4$.

1.5.2 Monopod Decompositions

Finally we consider the decomposition described in Theorem 1.1.c. This decomposition is obtained from a tripod decomposition $\mathcal{Y} := Y_0, \ldots, Y_r$, obtained by a sequence $\mathcal{F} := f_0, \ldots, f_r$ of faces of G in the same manner described in Section 1.3.1. However in this setting, the sequence $f_0, \ldots, f_r$ is *good* if, for each $i \in \{0, \ldots, r\}$ and each face f of $G_i := \bigcup_{j=0}^{i} Y_T(f_j)$, $V(f)$ contains vertices from at most 5 *legs* of tripods in $Y_0, \ldots, Y_i$. Under these conditions, Ueckerdt et al. [65] are able to show that the *monopod decomposition* $\mathcal{I}$ obtained by splitting each tripod Y_i into three upward paths yields a quotient graph $G/\mathcal{I}$ of treewidth at most 6.

As before we focus on the extreme case when $V(f)$ contains vertices from exactly 5 legs of tripods. Refer to Fig. 1.7. Following the same strategy used for the previous two decompositions, the set S in this case has size at most 5 and the face f_i corresponds to a node of $\overline{T}_0$ such that each component of $\overline{T}_0 - f_i$ contains at most 2 nodes in S. (This is always possible because $\lfloor 5/2 \rfloor = 2$.) Again, a suitable choice for f_i can be found in the model $\overline{T}_0'$ of $\mathcal{T}_0$ in constant time.

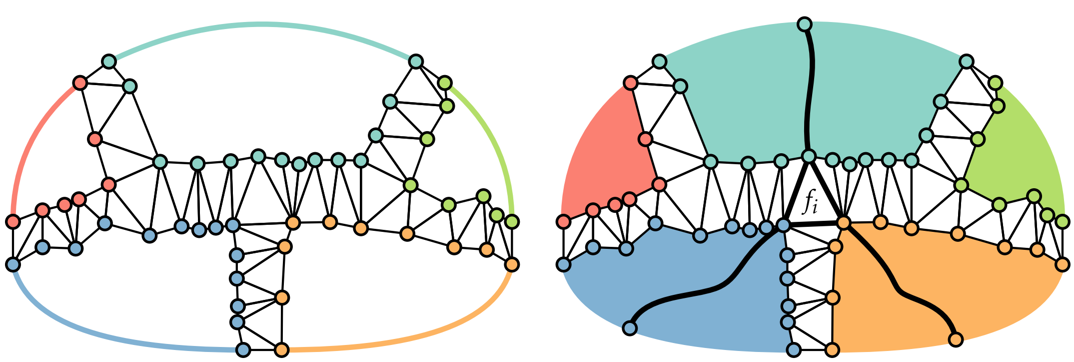


Fig. 1.7: The selection of a tripod by Ueckerdt et al. [65]

[11]In the degenerate case where C has no bichromatic edges, the input is any face of N incident to an edge of C.

Theorem 1.5. *There exists an $O(n)$ time algorithm that, given any n-vertex triangulation G and any rooted spanning tree T of G, produces a (G,T)-monopod decomposition $\mathcal{I}$ such that* $\mathrm{tw}(G/\mathcal{I}) \leq 6$.

Acknowledgement

This research was initiated at the BIRS 21w5235 Workshop on Graph Product Structure Theory, held November 21–26, 2021 at the Banff International Research Station. The authors are grateful to the workshop organizers and participants for providing a stimulating research environment. We are especially grateful to Vida Dujmović for sharing Theorem 1.1.b with us.

Chapter 2

On Separating Path and Tree Systems in Graphs

Ahmad Biniaz[1] Prosenjit Bose[2] Jean-Lou De Carufel[3]
Anil Maheshwari[2] Babak Miraftab[2] Saeed Odak[3]
Michiel Smid[2] Shakhar Smorodinsky[4] Yelena Yuditsky[5]

Abstract. We explore the concept of separating systems of vertex sets of graphs. A separating system of a set X is a collection of subsets of X such that for any pair of distinct elements in X, there exists a set in the separating system that contains exactly one of the two elements. A separating system of the vertex set of a graph G is called a vertex-separating path (tree) system of G if the elements of the separating system are paths (trees) in the graph G. In this chapter, we focus on the size of the smallest vertex-separating path (tree) system for different types of graphs, including trees, grids, and maximal outerplanar graphs.

2.1 Introduction

Given a set X, a collection $\mathcal{F}$ of subsets of X is called a *weakly separating system* or, simply, a *separating system* of X if for every pair of distinct elements, $a, b \in X$, there exists a set $F \in \mathcal{F}$ such that F separates a and b, that is, F contains exactly one of a and b. We refer to the smallest size of a separating system as the *separation number* of X.

Rényi [56] initiated this notion of separation in 1961. In fact, the separation number of a set of size n is $\lceil \log n \rceil$. By restricting the set X and enforcing some conditions on the elements of $\mathcal{F}$, several interesting variants of separating set system problems have been studied in the literature [7, 13, 14, 22, 51, 59, 67–69]. For example, when the

[1]School of Computer Science, University of Windsor
[2]School of Computer Science, Carleton University
[3]School of Electrical Engineering and Computer Science, University of Ottawa
[4]Department of Mathematics, Ben-Gurion University of the Negev
[5]Département de Mathématique, Université libre de Bruxelles

set X represents the edge set of a given n-vertex connected graph G, an *edge-separating path system* of G is defined as a collection of paths in G that separates the edges of G, that is, the elements of the separating system of X are paths in G. Falgas-Ravry et al. [33] conjectured that the smallest size of an edge-separating path system of G is independent of the size of the edge set and it is linear in the size of the vertex set, i.e. $O(n)$; this has been proven recently in [14]. Also, geometric versions of this problem, when the set X is an arbitrary point set in the plane and the separating sets are geometric objects, like circles and convex sets, have been studied in [39] and [40].

In this chapter, we focus on separating the vertex set of a graph with paths and trees. Let G be a graph and $X \subseteq V(G)$ be a set of vertices of G. A collection of distinct paths $\mathcal{S} := \{\Pi_1, \Pi_2, \Pi_3, \ldots, \Pi_k\}$ in G is called a *separating path system* of X if for every pair of distinct vertices, $u, v \in X$, there exists an $\ell \in \{1, 2, 3, \ldots, k\}$ such that Π_ℓ contains exactly one of u and v. We define a *vertex-separating path system* of the graph G to be a separating path system of $V(G)$. We are interested in the size of the smallest vertex-separating path system of the graph G, and we denote this number by $f(G)$. Analogously, we define a *vertex-separating tree system* of G, when the elements of the separating set $\mathcal{S}$ are tree subgraphs of G, and we denote the size of the smallest vertex-separating tree system of G by $f_t(G)$.

To the best of our knowledge, there exist only a few results in the literature introducing and studying vertex-separating path systems. Foucaud and Kovše [36] studied this parameter in the context of identifying codes, and they provide optimal vertex-separating path systems for path and cycle graphs. They also present the first upper and lower bounds for $f(T)$ when T is a tree. Recently, Arrepol et al. [5], among other variants, studied vertex-separating path systems of random graphs, and they also improved the upper and lower bounds of $f(T)$ when T is a tree. In Section 2.4, we briefly review the known bounds for trees. We then present a tight lower bound for the size of the smallest vertex-separating path system. In fact, we show that $f(T) \geq \frac{n}{4}$ for every n-vertex tree.

In Section 2.5, we focus on separating the vertices of an n by n grid using $O(\log n)$ paths, where $n \geq 2$. In a related study, Honkala et al. [45] use cycles to separate the vertices of a torus. Notably, Rosendahl [59] also studies the same parameter as discussed in [45] but for higher dimensions. Additionally, Rosendahl [59] analyze the precise values of $f(K_{n,n})$ and $f(K_{3,n})$, where $n \in \mathbb{N}$. In Section 2.3, by utilizing an old result by Katona [48], we present a tight asymptotic bound for the value of $f(K_{m,n})$, where m and n are arbitrary positive integers.

In addition to the aforementioned results, this chapter also includes a tight upper bound for the value of $f(G)$ when G is in the class of maximal outerplanar graphs. Moreover, we show that a $K_{2,t}$-minor-free graph with a high-degree vertex requires a polynomial-sized vertex-separating path system for any constant t. Next, we focus on the size of optimal vertex-separating tree systems and we prove that every n-vertex graph with radius r has a vertex-separating tree system of size at most $r + 2\log n + 1$.

The rest of this chapter is organized as follows: In section 2, we start with pre-

Graph class	Lower Bound	Upper Bound	Ref.
n-vertex complete graph, K_n	$\lceil \log n \rceil$	$\lceil \log n \rceil$	[56]
d-dimensional hypercube, Q_d	d	d	[36]
n-vertex path and cycle, P_n, C_n	$\lceil \frac{n}{2} \rceil$	$\lceil \frac{n}{2} \rceil$	[36]
Complete bipartite graph, $K_{m,n}$	$\Omega(\frac{n}{m} \cdot \frac{\log n}{\log(1+n/m)})$	$O(\frac{n}{m} \cdot \frac{\log n}{\log(1+n/m)})$	[48], Proposition 2.1
$m \times n$- grid graph, $G_{m,n}$ $(m,n \geq 2)$	$\lceil \log m + \log n \rceil$	$2\lceil \log m \rceil + 2\lceil \log n \rceil$	Theorem 2.4
Erdős–Rényi random graph $G(n,p)$, $p \geq (2\ln n + \omega(\ln\ln n))/n$	$\lceil \log n \rceil$	w.h.p. $\lceil \log n \rceil + 1$	[5]
Erdős–Rényi random graph $G(n,p)$, $p \leq (\ln n - \omega(\ln\ln n))/n$	w.h.p. $\omega(\log n)$	$O(n)$	[5]
n-vertex tree	$\frac{n}{4}$	$\frac{2n}{3} + O(1)$	[36], Theorem 2.3
n-vertex maximal outerplanar graph	$\Omega(\log n)$	$\frac{n}{4} + O(1)$	Theorem 2.5
n-vertex $K_{2,t}$-minor-free graph with a vertex of degree $\Omega(n^{\delta})$	$\Omega(n^{\delta/(t+1)})$	$\frac{2n}{3} + O(1)$	Theorem 2.7

Table 2.1: Summary of previous and new results on the value of $f(G)$ (w.h.p. stands for with high probability).

liminaries and some simple observations related to vertex-separating path systems. In Section 2.3, Section 2.4, Section 2.5, and Section 2.6, we will discuss vertex-separating path systems of complete bipartite graphs, trees, grids, and maximal outerplanar graphs, respectively (see Table 2.1). Section 2.7 will establish a sufficient condition that guarantees a polynomial-sized lower bound for the size of vertex-separating path systems in certain classes of graphs. Section 2.8 studies vertex-separating tree systems and compares this variant with vertex-separating path systems. Finally, we discuss some open problems in Section 2.9.

2.2 Preliminaries

For integers $0 \leq a \leq b$, put $[a] := \{x | x \in \mathbb{Z}$ and $1 \leq x \leq a\}$ and $[a,b] := \{x | x \in \mathbb{Z}$ and $a \leq x \leq b\}$. Throughout this chapter, we use standard graph theoretic terminology as used in the textbook by Diestel [26]. All graphs discussed here are connected, simple, finite, and have at least 4 vertices. We denote the vertex set and edge set of a graph G by $V(G)$ and $E(G)$, respectively. We say that a subgraph G' of a graph G *spans* a set $S \subseteq V(G)$ if $S \subseteq V(G')$.

A *path* in G is a sequence of distinct vertices $v_0, v_1, \ldots, v_r$ with the property that $\{v_{i-1}, v_i\} \in E(G)$, for each $i \in [r]$. The *endpoints* of such a path are the vertices v_0 and v_r. The *length* of a path is the number of edges in the path. A path of length zero is called a *trivial path*. If v is the only vertex in a trivial path Π, we say v *creates* Π. A vertex $v \in V(G)$ is called a center of the graph G if the largest distance of v to other vertices in $V(G) \setminus \{v\}$ is minimal (a center vertex may not be unique).

An $m \times n$ *grid* $G_{m,n}$ is a graph with vertex set $V(G_{m,n}) := \{0,1,2,\dots,m-1\} \times \{0,1,2,\dots,n-1\}$ and edge set $E(G_{m,n}) = \{(i,j)(i',j') | 0 \leq i,i' < m, 0 \leq j,j' < n \text{ and } |i-i'|+|j-j'| = 1\}$. For a vertex $(i,j) \in V(G_{m,n})$, we define the projection functions as $\pi_x(i,j) = i$ and $\pi_y(i,j) = j$. Each *column* of $G_{m,n}$ consists of vertices with the same π_y value, that is the vertex set $\{0,1,2\dots,m-1\} \times \{j\}$ for some fixed $j \in \{0,1,2,\dots,n-1\}$. Similarly, each *row* of $G_{m,n}$ consists of vertices with the same π_x value, that is the vertex set $\{i\} \times \{0,1,2,\dots,n-1\}$ for some fixed $i \in \{0,1,2,\dots,m-1\}$. A set C of columns (rows) is consecutive if $G_{m,n}[\cup C]$ is connected.

Let G be a graph and let S be a non-empty subset of $V(G)$. A *labeling* of S is a function $\psi : S \to [0, 2^{\lceil \log |S| \rceil} - 1]$. This labeling is called *nice* if for each $1 \leq i \leq \lceil \log |S| \rceil$, the graph induced by $(V(G) \setminus S) \cup S_i$ contains a path that spans the set $S_i := \{v \in S \mid \text{the } i\text{-th bit in the binary representation of } \psi(v) \text{ is } 1\}$.

Theorem 2.1 ([56]). *Let G be a graph and let S be a non-empty subset of $V(G)$. Then S has a separating path system of size $\lceil \log |S| \rceil$ if and only if S has a nice labeling.*

Let n be a positive integer. Since every induced subgraph of K_n has a spanning path, by taking $S = V(G)$ in Theorem 2.1, we have $f(K_n) = \lceil \log n \rceil$ (Note that any labeling of $V(G)$ is nice). This, in turn, implies that for any graph G with n vertices, $f(G) \geq \lceil \log n \rceil$. If we are aiming to cover $V(G)$ using the same set of paths in the separating system, we need at least $\lceil \log(n+1) \rceil$ paths.[6] By Theorem 2.1, we have that $f(Q_k) = k$, where Q_k is the k-dimensional hypercube [36]. As we will be referring to them in the sequel, we state the value of the parameter f for paths and cycles.

Observation 2.1. *([36, Theorem 15]) For every integer $n \geq 3$, $f(P_n) = f(C_n) = \lceil \frac{n}{2} \rceil$, where P_n and C_n denote an n-vertex path and cycle, respectively.*

2.3 Complete Bipartite Graphs

To initiate our study, we consider separating path systems of the vertices of complete bipartite graphs $K_{m,n}$. It is worth mentioning that vertex-separating cycle systems of $K_{m,n}$ have been studied in [59]. Let m and n be two positive integers with $m \leq n$. Note that the length of the longest path in $K_{m,n}$ is $2m$. Katona [48] provided bounds for the size of the smallest separating set system of $[n]$ where each set in the separating system has size at most $1 \leq k \leq n$. In fact, he showed that if $\tau(n,k)$ is the size of the smallest such separating set system, then $\frac{n}{k} \cdot \frac{\log n}{\log(en/k)} \leq \tau(n,k) \leq \frac{n}{k} \cdot \frac{\log 2n}{\log(n/k)}$. From this result, we note that $f(K_{m,n}) = \Theta(\frac{n}{m} \cdot \frac{\log n}{\log(1+n/m)})$. Since our construction is somewhat simpler, we provide a different proof for the upper bound of $f(K_{m,n})$.

Proposition 2.1. $f(K_{m,n}) = O(\frac{n}{m} \cdot \frac{\log n}{\log(1+n/m)})$, *for integers $n \geq m > 0$.*

[6] This is equivalent to not using the label 0 for any vertex in $V(G)$ (cf. [36, Proposition 2]).

Proof. We build a vertex-separating path system of the given size. Let L refer to the m vertices on the smaller part of $K_{m,n}$ and R refer to the other part of size n. For subsets $X \subseteq L$ and $Y \subseteq R$, we denote the induced subgraph of $K_{m,n}$ on $X \cup Y$ by $K(X,Y) := K_{m,n}[X \cup Y]$. In order to separate the vertices of L from R, consider a subset $R' \subseteq R$ of size m. Partition L into two almost equal size sets, L_1 and L_2. Similarly, partition R' into R'_1 and R'_2. By adding four paths covering the vertex sets of $K(L_1, R'_1)$, $K(L_1, R'_2)$, $K(L_2, R'_1)$, and $K(L_2, R'_2)$, we separate the vertices of L from the vertices of R.

Next, if $m = n$, then Theorem 2.1 implies that $f(K_{m,m}) = \Theta(\log m)$. More precisely, we separate the vertices of L from each other using a nice labeling of the vertices L. That is we assign labels $0, 1, \ldots, m-1$ to the vertices of L in arbitrary order. For each $1 \le i \le \lceil \log m \rceil$, using vertices in R, we create a path that spans only those vertices in L whose i-th bit in the binary representation of their labels is 1. Similarly, we separate the vertices of R from each other.

Now assume $n > m$. First, similar to the case $K_{m,m}$, using $O(\log m)$ paths, we separate the vertices in L from each other. It remains to separate the vertices within R from each other. To achieve this, define $K := \lceil \frac{n}{m} \rceil$ and consider an auxiliary tree $\mathcal{T}$ having the elements of R at its leaves, in which each internal node has K children. We refer to the vertices of $\mathcal{T}$ as nodes. Note that the height of $\mathcal{T}$ is $O(\frac{\log n}{\log K})$. In order to simplify the explanation, we assume that n is a power of K (otherwise, each internal level of $\mathcal{T}$ will contain at most one node with less than K children). For each node u in this tree, let S_u denote the set of elements of R that are stored in the subtree rooted at u.

For each internal level ℓ in the tree, and for each $i = 1, 2, 3, \ldots, K$, let $R(\ell, i)$ be the union of all sets S_u, where u ranges over the i-th child of all nodes at level ℓ. Note that the size of $R(\ell, i)$ is at most m. We add a path in $K(L, R(\ell, i))$ that spans $R(\ell, i)$ to the separating path system.

Now, we show that the set of selected paths in the previous paragraph separates the vertices of R from each other. Let r and r' be two distinct vertices in R. Let v and v' be the leaves of $\mathcal{T}$ that store r and r', respectively. Let u be the lowest common ancestor of v and v' in $\mathcal{T}$, and let ℓ be the level of u. Let i be such that r is in the subtree rooted at the i-th child of u, and define i' similarly with respect to r'. Since $i \neq i'$, the vertices r and r' are separated by the spanning path in $K(L, R(\ell, i))$.

In this construction, $\mathcal{T}$ has $O(\frac{\log n}{\log K})$ levels, and in each level we select $O(K)$ paths. Therefore, the total number of paths used to separate the vertices of $K_{m,n}$ is $O(\log m + K \cdot \frac{\log n}{\log K}) = O(\log m + \frac{n}{m} \cdot \frac{\log n}{\log(n/m)}) = O(\frac{n}{m} \cdot \frac{\log n}{\log(1+n/m)})$, since $K = \lceil \frac{n}{m} \rceil$. □

As a corollary of Proposition 2.1, one can show that the value of the parameter $f(G)$ for n-vertex graphs G, asymptotically, can be as large as any sub-linear polynomial on n.

Corollary 2.1. *Let $0 < \delta \le 1$ be a real number. For any positive integers m and n, with $m = n^{1-\delta}$, $f(K_{m,n}) = \Theta(n^{\delta})$.*

2.4 Trees

As mentioned in the introduction, the smallest size of vertex-separating path systems of trees has been studied by Foucaud and Kovše [36] and Arrepol et al. [5]. In particular, Foucaud and Kovše [36] show that $f(T) \leq \frac{2n}{3} + O(1)$, for any n-vertex tree T. Moreover, they show that for the star $K_{1,n-1}$, $f(K_{1,n-1}) = \frac{2n}{3} + O(1)$. Thus $K_{1,n-1}$ serves as a matching lower bound example. Arrepol et al. [5] provided bounds as a function of the number of degree one vertices and degree two vertices in T, denoted by A_1 and A_2 respectively, and the number of special bare paths $\mathcal{I}$ — the number of paths of length at least two in T such that its two endpoints have a degree at least three and all the other vertices on the path have degree two. Their bound for trees reads as follows:

Theorem 2.2. *([5, Theorem 3.4]) Let T be a tree with A_1 degree one vertices, A_2 degree two vertices, and $\mathcal{I}$ special bare paths. Then,*

$$\max\left\{\left\lceil \frac{2A_1 + A_2 - \mathcal{I}}{3} \right\rceil, \left\lceil \frac{A_1 + A_2 - \mathcal{I}}{2} \right\rceil\right\} \leq f(T) \leq \frac{2A_1}{3} + \frac{A_2 - \mathcal{I}}{2} + O(1).$$

In this section, we present a tight lower bound for the size of an optimal vertex-separating path system of any n-vertex tree as a function of n. Our lower bound does not follow directly from Theorem 2.2. For example, one can check that if T is an n-vertex tree obtained from a binary tree where every edge between two vertices of degree three is subdivided once, then Theorem 2.2 implies that $f(T) \geq \frac{2n}{9}$.

Theorem 2.3. *Let T be an n-vertex tree. Then $f(T) \geq \frac{n}{4}$. Moreover, this lower bound is tight (up to an additive constant).*

We start with a simple observation relating the low-degree vertices of a graph with the endpoint of paths in an arbitrary vertex-separating path system.

Observation 2.2. *Let G be a graph and let $\mathcal{S}$ be a vertex-separating path system of G.*

(I) Let A_1 be the number of degree one vertices in G. At least $A_1 - 1$ degree one vertices are the endpoints of some path in $\mathcal{S}$.

(II) Let $\Pi \in \mathcal{S}$ be a non-trivial path such that both endpoints, say u and v, of Π are degree one vertices of G. There exists a path $\Pi' \in \mathcal{S}$, $\Pi' \neq \Pi$, such that Π' contains exactly one of u and v as an endpoint.

(III) Let u and v be two adjacent degree two vertices of G. There is a path in $\mathcal{S}$ that ends in exactly one of u and v.

(IV) Let u be a degree one vertex of G that is adjacent to a degree two vertex v. If there is no trivial path containing u in $\mathcal{S}$, there exists a path in $\mathcal{S}$ with v as an endpoint that does not contain u.

We will use this observation to prove a tight lower bound for $f(T)$ where T is an arbitrary n-vertex tree.

Proposition 2.2. *Let $n > 2$ be an integer and T be an n-vertex tree. Then $f(T) \geq \frac{n}{4}$.*

Proof. Let T be an arbitrary n-vertex tree. Let A_1, A_2, and $A_{\geq 3}$ be the number of degree one, degree two, and degree at least three vertices of T, respectively. Let $\mathcal{S}$ be a vertex-separating path system of T. If T is a path then by Observation 2.1, $|\mathcal{S}| = \lceil \frac{n}{2} \rceil \geq \frac{n}{4}$. Therefore, we can assume T is rooted at a vertex of degree at least three. Let ϕ be the number of leaves of T that create trivial paths in $\mathcal{S}$. We prove the lower bound by induction on pair (ϕ, n) in lexicographic order. For the base case, assume that $\phi = 0$.

We say that an edge $e = \{x, y\}$ in T is *good* if x is a degree two vertex of T and y is the only child of x such that y is either a vertex of degree at most two or there is no path in $\mathcal{S}$ that contains y but does not contain x.

Claim 2.1. *For each good edge $e \in E(T)$, there exists a distinct endpoint of some path in S located on a degree two vertex of e.*

Proof. Let $\{x, y\} \in E(T)$ be a good edge where x is a vertex of degree two and y is the child of x in T. We have three cases:

(I) y is a vertex of degree one. By Observation 2.2 (IV) and since there is no trivial path in $\mathcal{S}$ ($\phi = 0$), there is a path $\Pi \in \mathcal{S}$ such that one endpoint of Π is on the degree two vertex x and Π does not contain y.

(II) y is a vertex of degree two. By Observation 2.2 (III), there is a path $\Pi \in \mathcal{S}$ that ends in exactly one of x and y; in fact, Π separates x and y.

(III) y is a vertex of degree at least three and every path of $\mathcal{S}$ that contains y also contains x. Then, in order to separate x and y, there exists a path in $\mathcal{S}$ that ends at x not containing y.

For a good edge $e \in E(T)$, according to these three cases, an endpoint of some path in $\mathcal{S}$ is at a degree two endpoint of e. Let u be a vertex of degree two, then each non-trivial path with one endpoint at u contains one of the neighbors of u. Therefore, an endpoint of a non-trivial path in $\mathcal{S}$ does not separate endpoints of two good edges. If a degree two vertex is a trivial path in $\mathcal{S}$, we consider two endpoints for that trivial path. Therefore, each good edge has a distinct corresponding endpoint of a path from $\mathcal{S}$. $\triangle$

Claim 2.2. *The tree T has at least $A_2 - |S|$ good edges.*

Proof. Let u be a vertex of degree two, and v be its only child in T. If v has degree at most two, then the edge $\{u, v\}$ is a good edge. For each path $\Pi \in \mathcal{S}$, only the parent of one vertex in $V(\Pi)$ is not in Π. Hence, there are at most $|\mathcal{S}|$ degree-two vertices

$u \in V(T)$ such that its child v in T has degree at least three, and there is a path in $\mathcal{S}$ that contains v but does not contain u. Therefore, there are at least $A_2 - |S|$ good edges in T. △

Let L be the set of degree-one vertices in T, and note that $|L| = A_1$. Let D be the set of degree two vertices $v \in V(T)$, for which there exists a path Π in $\mathcal{S}$ that has v as an endpoint. We use a charging scheme to prove the lower bound. We assign a charge of 1 per path in $\mathcal{S}$. Then we consider the following discharging rules. Let $\Pi \in \mathcal{S}$ be a path with u and v as endpoints.

(I) If $u, v \in L \cup D$, then both u and v get $\frac{1}{2}$ charge from Π.

(II) If $u \in L \cup D$ and $v \notin L \cup D$, then u gets 1 charge from Π.

After discharging, every vertex in D gets at least $\frac{1}{2}$ charge. Denote by ch_D the total charge stored at the vertices of D. Recall that for each good edge $e \in E(T)$, there exists at least one endpoint of some path in S located on a degree two vertex of e. Note that these endpoints are distinct. Therefore, ch_D is at least $\frac{1}{2}(A_2 - |S|)$.

Let X be the set of all vertices in L such that there exists a path $\Pi \in \mathcal{S}$ with one endpoint in X and one endpoint in D. Let $r = |X|$. By definition of X, $\frac{r}{2}$ is another lower bound for ch_D, that is $ch_D \geq \frac{r}{2}$. Then by Observation 2.2 (II, IV), at least $\frac{2}{3}(|L| - r - 1)$ charge is located on vertices in $L \setminus X$ (cf. [36, Proposition 12]). Hence,

$$|\mathcal{S}| \geq \frac{2}{3}(|L| - r - 1) + \frac{1}{2}r + ch_D = \frac{2}{3}A_1 - \frac{1}{6}r + ch_D - \frac{2}{3}. \qquad (\spadesuit)$$

We consider two cases:

(a) $r \geq A_2 - |\mathcal{S}|$: By ($\spadesuit$) and $ch_D \geq \frac{r}{2}$, we have

$$|\mathcal{S}| \geq \frac{2}{3}A_1 - \frac{1}{6}r + ch_D - \frac{2}{3} \geq \frac{2}{3}A_1 + \frac{1}{3}r - \frac{2}{3} \geq \frac{2}{3}A_1 + \frac{1}{3}(A_2 - |\mathcal{S}|) - \frac{2}{3}.$$

(b) $r < A_2 - |\mathcal{S}|$: By ($\spadesuit$) and $ch_D \geq \frac{1}{2}(A_2 - |\mathcal{S}|)$, we have

$$|\mathcal{S}| \geq \frac{2}{3}A_1 - \frac{1}{6}r + ch_D - \frac{2}{3} \geq \frac{2}{3}A_1 - \frac{1}{6}(A_2 - |\mathcal{S}|) + \frac{1}{2}(A_2 - |\mathcal{S}|) - \frac{2}{3} \geq \frac{2}{3}A_1 + \frac{1}{3}(A_2 - |\mathcal{S}|) - \frac{2}{3}.$$

In both cases, we have $|\mathcal{S}| \geq \frac{2}{3}A_1 + \frac{1}{3}(A_2 - |\mathcal{S}|) - \frac{2}{3}$, that is

$$|\mathcal{S}| \geq \frac{1}{2}A_1 + \frac{1}{4}A_2 - \frac{1}{2} = \frac{1}{4}A_1 + \frac{1}{4}A_2 + \frac{1}{4}A_1 - \frac{1}{2} \geq \frac{1}{4}A_1 + \frac{1}{4}A_2 + \frac{1}{4}(A_{\geq 3} + 2) - \frac{1}{2} = \frac{n}{4}.$$

For the induction step, assume $\phi > 0$ and let $v \in V(T)$ be a leaf that creates a trivial path in $\mathcal{S}$. We consider two cases:

Let $u \in V(T)$ be the only neighbor of v in T. Assume u creates a trivial path in $\mathcal{S}$ and the edge (v,u) is a path in $\mathcal{S}$. Let $\mathcal{S}'$ be a set obtained by removing the trivial path corresponding to v from $\mathcal{S}$. Observe that $\mathcal{S}'$ is a vertex-separating path system of T. By induction hypothesis, we have $|\mathcal{S}| > |\mathcal{S}'| \geq \frac{n}{4}$.

Otherwise, let T' be the tree that is obtained from T by removing v. Let $\mathcal{S}'$ be a vertex-separating path system of T' obtained from $\mathcal{S}$ by excluding trivial path at v and removing v from every path in $\mathcal{S}$. Since the number of trivial paths in $\mathcal{S}'$ is not greater than ϕ and the number of vertices of T' is one less than the number of vertices of T, we have $|\mathcal{S}'| \geq \frac{n-1}{4}$. Therefore $|\mathcal{S}| \geq \frac{n-1}{4} + 1 > \frac{n}{4}$. □

In order to prove the tightness, we build infinitely many trees T such that their vertices can be separated using $\frac{|V(T)|}{4} + O(1)$ paths.

Proposition 2.3. *There are infinitely many trees T with $f(T) = \frac{|V(T)|}{4} + O(1)$.*

Proof. Consider a planar drawing of the complete binary tree B_h of height $h-1$, with $2^h - 1$ vertices, for a positive integer h. We label the leaves of B_h with the numbers $0, 1, 2, \ldots, 2^{h-1} - 1$ from left to right. We construct a vertex-separating path system, denoted by $\mathcal{S}$, for B_h. $\mathcal{S}$ contains the unique path between the leaves labeled i and $i+1$, for each i with $0 \leq i < 2^{h-1}$ and $i \not\equiv 0 \pmod 4$, and the unique path between the leaves labeled 0 and $2^{h-1} - 1$ (see Fig. 2.1). If we connect the leaves labeled with consecutive numbers by an edge, we obtain a planar graph where each path corresponds to a face of this graph. Due to the properties of the complete binary tree, every pair of non-adjacent vertices in this planar graph is incident to at most one common internal face. Using a simple inductive argument, it can be shown that $\mathcal{S}$ is indeed a vertex-separating path system of B_h. Moreover, every pair of edges is separated from each other, meaning that every pair of distinct edges e_1 and e_2 in $E(T)$ are separated by a path in $\mathcal{S}$ (Except the edges incident to the root).

We number the levels in B_h from 1 to h, where the root is at level one. Next, by adding $2^{h-2} - 2$ vertices, we subdivide every edge with both endpoints at the levels at most $h-2$. Then, consider all the edges with one endpoint at level $h-2$ and another endpoint at level $h-1$ in the left to right order. We subdivide every second one of such edges. This step will add 2^{h-3} vertices to the graph (see Fig. 2.1). We refer to the tree obtained after these modifications, with $2^h + 2^{h-2} + 2^{h-3} - 3$ vertices, as B'_h. Let $\mathcal{S}'$ be the set of paths in B'_h with the same endpoints as the paths in $\mathcal{S}$. To show that $\mathcal{S}'$ is a vertex-separating path system of B'_h, we only need to consider the separation of subdivision vertices (the separation from the root of B'_h is a special case and will be handled separately). Every subdivision vertex v is on two different paths in $\mathcal{S}'$ and is incident to two vertices of degree three, say v_1 and v_2. Therefore, by construction, there are three different paths going through each neighbor of v, while exactly two of them pass through v. To separate v from other vertices in $V(B'_h) \setminus \{v_1, v_2\}$, we consider the cases where u is a subdivision vertex, an internal vertex of B_h, or a leaf of B'_h. In the former two cases, we use the fact that there is an edge incident to u that is separated

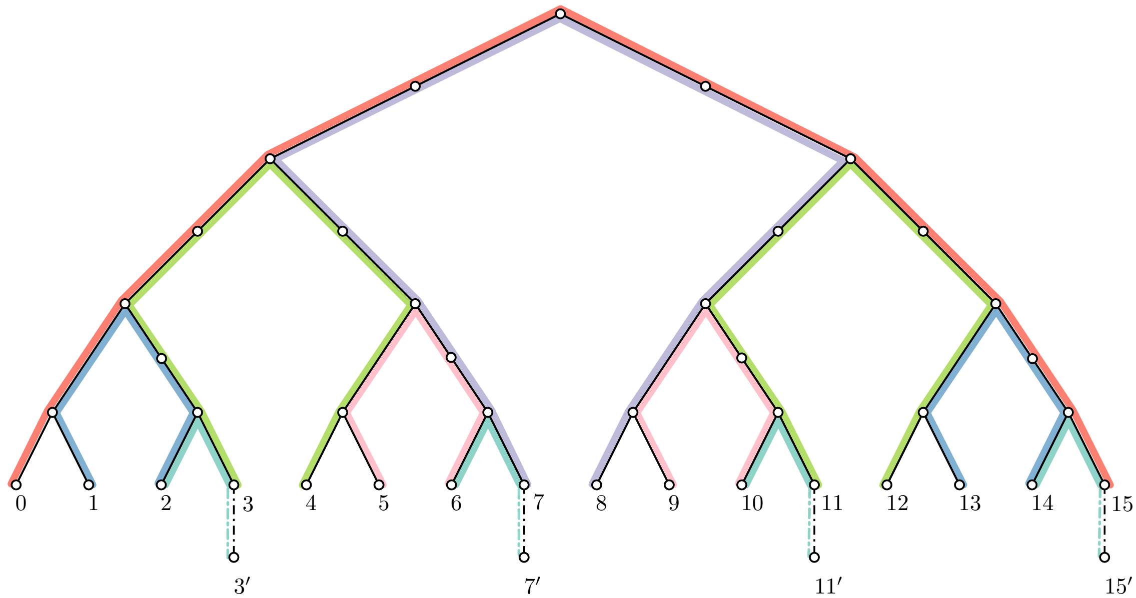


Fig. 2.1: The tree T_5 obtained from the complete binary tree of height 4.

by a path in $\mathcal{S}$ from the edge on which v is located. For the latter case, we note that at least one of the two paths that go through the edge $\{v_1, v_2\}$ in B_h does not cover the vertex u.

For each $0 \le \ell < 2^{h-1}$ where $\ell \bmod 4 = 2$, according to our construction, there exists a path of length two between the leaf labeled ℓ and $\ell + 1$. As the final step of the construction, we create the tree T_h by adding an edge to each leaf located at $\ell + 1$. We extend the path of length two between ℓ and $\ell+1$ in $\mathcal{S}'$ to cover the new edge (indicated by dashed edges in Fig. 2.1). Once again, based on our construction, the extended paths of length three will separate the new vertex from the rest of the vertices. This final step adds 2^{h-3} vertices to obtain the final tree, resulting in a total of $3 \cdot 2^{h-1} - 3$ vertices in T_h. As mentioned before, the only vertex that is not separated from its two neighbors is the root of the binary tree, and this can be solved by adding two extra paths.

Now if $\mathcal{P}$ denotes our vertex-separating path system of T_h, then we have

$$|\mathcal{P}| \le 3 \cdot 2^{h-3} + 2 < \frac{1}{4} \cdot |V(T_h)| + 3 = \frac{1}{4} \cdot |V(T_h)| + O(1).$$

□

By combining the previous two propositions, one can establish the proof of Theorem 2.3.

2.5 Grid Graphs

In this section, we study vertex-separating path systems of grids. Vertex-separating cycle systems of the $m \times n$ torus have been studied in [45] and [59]. Here, we separate the vertices of an $m \times n$ grid, $G_{m,n}$, using paths. We start with a simple observation about the existence of a Hamiltonian path in grid graphs.

Observation 2.3. *Let m and n be two positive integers. The grid $G_{m,n}$ has a Hamiltonian path with both endpoints on the last row.*

Now we state the main theorem of this section regarding the value of $f(G_{m,n})$ when $m, n \geq 2$.

Theorem 2.4. *Let $m, n \geq 2$ be two integers. Then $f(G_{m,n}) \leq 2\lceil \log m \rceil + 2\lceil \log n \rceil$.*

Proof. Let $m, n \geq 2$ be two integers. Recall that we represent the vertex set of $G_{m,n}$ by $V(G_{m,n}) = \{0, 1, 2, \ldots, m-1\} \times \{0, 1, 2, \ldots, n-1\}$. To begin with, we separate the vertices within different columns of the subgrid induced by the first $m-1$ rows of $G_{m,n}$ from each other by a separating path system of size $O(\log n)$. With a similar idea as in Theorem 2.1, we find a nice labeling of the columns. For each $2 \leq i \leq \lceil \log n \rceil$, let $A_i = \{(x, y) \in V(G_{m,n}) \mid x < m-1$ and the i-th bit in the binary representation of y is $1\}$. We construct a spanning path of A_i, say Π_i, such that Π_i does not intersect the first $m-1$ rows of $G_{m,n}$ on vertices other than vertices in A_i.

Note that, for each $2 \leq i \leq \lceil \log n \rceil$, each component of $G[A_i]$ consists of at least two consecutive columns of G. By Observation 2.3, we consider a Hamiltonian path with endpoints on the $(m-1)$-th row for each component of $G[A_i]$. We can merge the Hamiltonian path of these sub-grids using the m-th row. Therefore, Π_i as a spanning path of A_i exists (e.g. path Π_3 is depicted in Fig. 2.2). By construction, this set of paths will separate the vertices of every pair of columns, except for $\lfloor \frac{n}{2} \rfloor$ pairs of columns, namely $\{0, 1\}, \{2, 3\}, \{4, 5\}, \ldots, \{2\lceil \frac{n}{2} \rceil - 2, 2\lceil \frac{n}{2} \rceil - 1\}$.

For the purpose of separating these pairs of columns, we introduce the set $A_1 = \{(x, y) \in V(G_{m,n}) | x < m-1$ and $y \bmod 4 = 1$ or $2\}$. We then construct a spanning path Π_1 for the vertices of A_1 following the same procedure as described previously for $\Pi_2, \ldots, \Pi_{\lceil \log n \rceil}$ (refer to Fig. 2.2). Overall, the paths $\Pi_1, \Pi_2, \Pi_3, \ldots, \Pi_{\lceil \log n \rceil}$ separate every pair of vertices located in the first $m-1$ rows and in different columns. By repeating the same idea for the last $m-1$ rows, we can guarantee that with $2\lceil \log n \rceil$ paths, every pair of vertices $u, v \in V(G_{m,n})$ is separated if $\pi_y(u) \neq \pi_y(v)$.

Using an analogous construction, a set of paths with a size of $2\lceil \log m \rceil$ would separate vertices located in different rows. Since every pair of vertices in $V(G_{m,n})$ differs in their π_x or/and π_y values, these paths will form a vertex-separating path system of size at most $2\lceil \log m \rceil + 2\lceil \log n \rceil$.

□

2.6 Maximal Outerplanar Graphs

In this section, we explore vertex-separating path systems of maximal outerplanar graphs, i.e., graphs that are triangulations of convex polygons. The *inner dual* of a maximal outerplanar graph is its dual where the vertex corresponding to the outer face is

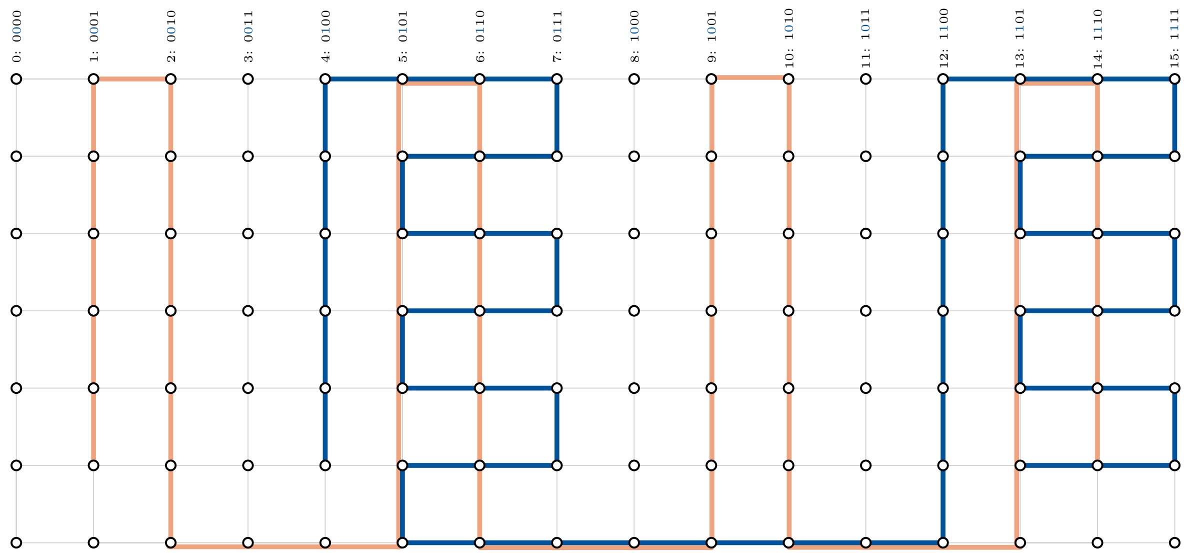


Fig. 2.2: Path Π_3 in blue colour and path Π_1 in orange colour in a 7×16 grid

removed. Theorem 2.4 implies that $f(G_{2,n}) = O(\log n)$. Hence, unlike trees, there are outerplanar graphs with a vertex-separating path system of size $O(\log n)$. The class of trees is a subclass of outerplanar graphs. Therefore, we cannot hope to improve the upper bound for the class of outerplanar graphs in general. However, since each 2-connected outerplanar graph G has a Hamiltonian cycle, using Observation 2.1, one would notice that $f(G) \leq \lceil \frac{n}{2} \rceil$. In fact, considering maximal outerplanar graphs leads to a further improvement in the upper bound. As a building block of maximal outerplanar graphs, we first consider the fan graph, F_n, on n vertices, that is, a path P_{n-1} on $n-1$ vertices and an apex vertex connected to all the vertices of the path.

Lemma 2.1. *Let F_n be the fan graph with n vertices. Then $f(F_n) = n/4 + O(1)$.*

Proof. Let $\mathcal{P}$ be a vertex-separating path system of F_n. Using $\mathcal{P}$, we obtain a vertex-separating path system, $\mathcal{P}'$, for P_{n-1}. For each path $\Pi \in \mathcal{P}$, if Π contains the apex vertex of F_n then we create at most two paths for $\mathcal{P}'$ from Π by removing the apex vertex, otherwise, we include Π in $\mathcal{P}'$. Since $\mathcal{P}$ is a separating path system for F_n, $\mathcal{P}'$ is a separating path system for P_{n-1}. By Observation 2.1, we know $\lceil \frac{n-1}{2} \rceil = f(P_{n-1}) \leq 2 \cdot f(F_n)$, therefore, $f(F_n) \geq \frac{n-1}{4}$.

To prove the upper bound, we construct a separating path system of size $\frac{n+10}{4}$ when $n \geq 6$. Let Π be the induced path of size $n-1$ in F_n. We split Π into sub-paths Π_l and Π_r of sizes $\left\lfloor \frac{n-1}{2} \right\rfloor$ and $\left\lceil \frac{n-1}{2} \right\rceil$, respectively. In order to separate the vertices of Π_l and Π_r, we include the path Π_l in the separating path system. Again by Observation 2.1, we construct a separating path systems $\mathcal{P}_l$ and $\mathcal{P}_r$ of sizes at most $\left\lceil \frac{|\Pi_r|}{2} \right\rceil \leq \frac{n+2}{4}$ for Π_l and Π_r, respectively. Since every pair of vertices $u \in \Pi_l$ and $v \in \Pi_r$ are already separated, we can merge the paths of $\mathcal{P}_l$ and $\mathcal{P}_r$ through the apex of F_n. Moreover, we add a single vertex path on the apex to separate the apex form the rest of the vertices in $V(F_n)$. By

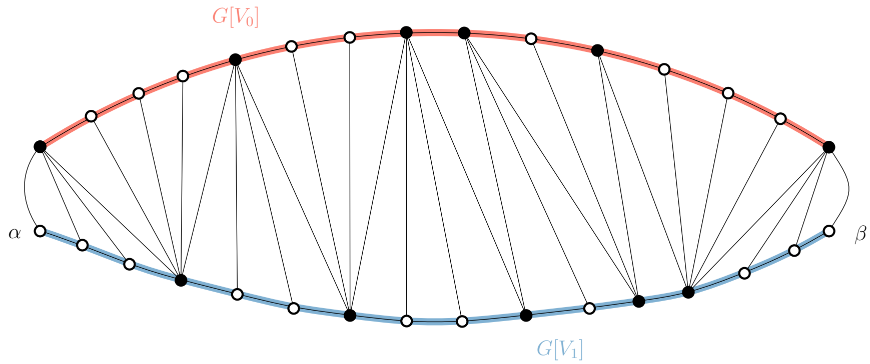


Fig. 2.3: An outerplanar graph in which the inner dual is a path. The black vertices represent the apex vertices of maximal fan subgraphs.

construction and the fact that $n \geq 6$, observe that this set of paths is a vertex-separating path system for F_n. Hence, $f(F_n) \leq \frac{n+2}{4} + 2 \leq \frac{n+10}{4}$. □

In Section 2.7, we confirm that, indeed, having just one high-degree vertex in an outerplanar graph is sufficient to establish a polynomial lower bound for the size of a vertex-separating path system. To extend the result of Lemma 2.1 to all maximal outerplanar graphs, we make the following observation.

Observation 2.4. *Let G be a maximal outerplanar graph such that the inner dual of G is a path. Then G can be decomposed into maximal induced fan subgraphs, $F_1, F_2, F_3, \ldots, F_k$, such that $G = \bigcup_{i=1}^{k} F_i$ and $|V(F_i) \cap V(F_j)| \leq 2$ for $1 \leq i < j \leq k$.*

Lemma 2.2. *Let G be an n-vertex maximal outerplanar graph such that the inner dual of G is a path, then $f(G) \leq \frac{n}{4} + O(1)$.*

Proof. Let $\mathcal{F} := \{F_1, F_2, F_3, \ldots, F_k\}$ be the set of all maximal fan subgraphs of G, as in Observation 2.4. Assume $k > 1$, otherwise, the result is implied from Lemma 2.1. For each $1 \leq i \leq k$, the vertex set of the graph F_i consists of an apex vertex a_i and vertices, $\mathcal{V}_i := \{v_{i,1}, v_{i,2}, v_{i,3}, \ldots, v_{i,n_i}\}$, along an induced path on the outer face of G in clockwise order. Note that $2 \leq n_i = |V(F_i)| - 1$ and a_i is the unique common neighbour of the vertices in $\mathcal{V}_i$. Moreover, if $1 \leq i < k$, $v_{i,n_i} = v_{(i+1),1}$.

Since the inner dual of G is a path, G contains exactly two vertices of degree two, say α and β. There are two different paths, $P_{\alpha\beta}$ and $P_{\beta\alpha}$, between α and β on the outer face of G. None of the vertices α and β is an apex vertex for any graph in $\mathcal{F}$ (See Fig. 2.3). We partition the set $\mathcal{F}$ into two sets of almost equal size $\mathcal{F}_0 = \{F_i \in \mathcal{F} | a_i \in P_{\alpha\beta}\}$ and $\mathcal{F}_1 = \{F_i \in \mathcal{F} | a_i \in P_{\beta\alpha}\}$ and define $V_0 := \bigcup_{F_i \in \mathcal{F}_0} \mathcal{V}_i$ and $V_1 := \bigcup_{F_i \in \mathcal{F}_1} \mathcal{V}_i$. V_0 and V_1 create a partition of $V(G)$ and $G[V_0]$ and $G[V_1]$ are two disjoint paths on the outer face of G.

For each $i \in \{0, 1\}$, let G_i be the fan graph obtained by contracting the connected subgraph $G[V_i]$ of G into a single vertex. We separate the vertices within V_i from each other by applying the result of Lemma 2.1 to G_{1-i}, where the subgraph $G[V_{1-i}]$ plays the role of the apex vertex in the construction explained in Lemma 2.1. To separate the vertices of V_0 and V_1 from each other, we only need to consider one extra path, namely $G[V_0]$ or $G[V_1]$[7]. So the total number of paths used to separate $V(G)$ is at most $f(G_0) + f(G_1) + 1 \leq \frac{n}{4} + O(1)$. □

Theorem 2.5. *Let G be an n-vertex maximal outerplanar graph for $n > 3$. Then $f(G) \leq \frac{n}{4} + O(1)$. Moreover, This upper bound is tight up to an additive constant.*

Proof. Let n be the number of vertices of G and k be the number of leaves of the inner dual of G. We prove this statement by induction on pair (k, n). For the base case, note that the inner dual of each maximal outerplanar graph contains at least two leaves and Lemma 2.2 proves the statement when the number of leaves is equal to two.

For the inductive step, if the inner dual of G has at most k leaves, where $3 \leq k \leq 19$, then we decompose G into constant number of maximal outerplanar graphs $G_1, G_2, G_3, \ldots, G_{k-1}$, such that $G = \bigcup_{i=1}^{k-1} G_i$ and the inner dual of G_i is a path and $|V(G_i) \cap V(G_j)| \leq 1$ for $1 \leq i < j < k$. For each $1 \leq i < k$, let $\mathcal{P}_i$ be the separating path system obtained by applying Lemma 2.2 to G_i. By construction, each $\mathcal{P}_i$ is a covering path system for G_i. Consider $\mathcal{P} = \bigcup_{i=1}^{k-1} \mathcal{P}_i$ as a separating path system of G. Since k is a constant, we have

$$f(G) \leq \sum_{i=1}^{k-1} f(G_i) \leq \sum_{i=1}^{k-1} \left(\frac{V(G_i)}{4} + O(1) \right) \leq \frac{n}{4} + O(1).$$

Now assume that the inner dual of G has $k \geq 20$ leaves. We use at most $\lceil \log(k+1) \rceil$ paths to separate the degree two vertices of G associated with these k leaves of the dual. Let S be the set of degree two vertices of G; thus, $|S| = k$. We are aiming to find a nice labeling for the vertices of S. The graph $G' = G[V(G) \setminus S]$ is a maximal outerplanar graph. We show that for each $A \subseteq S$, there exists a path in G that covers all vertices of A and no vertex of $S \setminus A$. Because no two vertices of degree two of G are adjacent, every degree two vertex of G is adjacent to two consecutive vertices on the outer face of G'. We construct such a path by considering a Hamiltonian path of G' and extending it to contain only the vertices of A. Therefore, we can apply the result of Theorem 2.1, to separate the vertices in S from each other by a separating path system of size $\lceil \log(k+1) \rceil$ (the plus one in the log function is to ensure that the constructed separating path system covers S). Observe that G' is a maximal outerplanar graph with a strictly smaller number of vertices and the number of leaves in its inner dual is no more than the number of leaves in the inner dual of G. We apply the induction hypothesis to the graph G'. Since $\frac{\lceil \log(k+1) \rceil}{k} \leq \frac{1}{4}$ for $k \geq 20$, we have

$$f(G) \leq f(G') + \lceil \log(k+1) \rceil \leq \frac{|V(G')|}{4} + O(1) + \lceil \log(|S|+1) \rceil$$

[7] In the proof of Theorem 2.5, we include both $G[V_0]$ and $G[V_1]$ in the vertex-separating path system. This guarantees that the resulting separating system covers $V(G)$.

$$\leq \frac{|V(G)|}{4} + O(1) = \frac{n}{4} + O(1).$$

Lemma 2.1 proves the tightness of this upper bound up to an additive constant. □

2.7 Graph Classes With Polynomial Lower Bound

This section will demonstrate that for every positive integer t, $K_{2,t}$-minor-free graphs with a high-degree vertex require a polynomial-sized vertex-separating path system.

To begin with, we introduce the Vapnik–Chervonenkis (VC) dimension of a set system and the relevant concepts. Let $(X,\mathcal{S})$ be a set system. We say that a subset $A \subseteq X$ is *shattered* by $\mathcal{S}$ if every subset of A can be expressed as the intersection of some $B \in \mathcal{S}$ with A. We define the *VC-dimension* of $(X,\mathcal{S})$ as the supremum of the sizes of all finite subsets of X that can be shattered by $\mathcal{S}$. We define the *shatter function* of a set system $(X,\mathcal{S})$ as:

$$\pi_{\mathcal{S}}(k) = \max_{Y \subseteq X, |Y|=k} |\{B \cap Y \mid B \in \mathcal{S}\}|.$$

In words, $\pi_{\mathcal{S}}(k)$ is the maximum possible number of distinct intersections of the sets of $\mathcal{S}$ with a k-element subset Y of X. It is bounded by the following lemma:

Lemma 2.3 ([53, 60, 62, 66])**.** *Let $(X,\mathcal{S})$ be a set system with VC-dimension d then $\pi_{\mathcal{S}}(k) \leq \Phi_d(k)$, where $\Phi_d(k) = \binom{k}{0} + \binom{k}{1} + \cdots + \binom{k}{d}$. In particular, for $k \geq d$ one has $\pi_{\mathcal{S}}(k) \leq (\frac{e}{d})^d \cdot k^d$, where e is Euler's number.*

The *dual set system* of $(X,\mathcal{S})$ is the set system $(\mathcal{S},\mathcal{X}^*)$, where $\mathcal{X}^* := \{\mathcal{S}_x | x \in X\}$ and $\mathcal{S}_x := \{B \in \mathcal{S} | x \in B\}$. The *dual shatter function* of the set system $(X,\mathcal{S})$ is the shatter function of the dual set system of $(X,\mathcal{S})$ and is represented by $\pi^*_{\mathcal{S}}(k)$. For a set system $(X,\mathcal{S})$, a set $B \in \mathcal{S}$ *crosses* a pair of elements $\{x,y\} \subseteq X$ if and only if exactly one element in $\{x,y\}$ is contained in B. In the terminology of this chapter, we say B separates $\{x,y\}$. A *matching* on the set X is a disjoint collection of pairs of elements of X. A *perfect matching* of X is a matching of size $\left\lfloor \frac{|X|}{2} \right\rfloor$. We define the *crossing number* of a matching M on the elements in X with respect to $\mathcal{S}$ to be the maximum number of pairs in M crossed by any set $B \in \mathcal{S}$. The following result illustrates the relationship between the dual shatter function and the crossing number of a matching on a set system.

Theorem 2.6 ([23],[43])**.** *Let $(X,\mathcal{S})$ be a set system with $|X| = n$ and dual shatter function $\pi^*_{\mathcal{S}}(k) = O(k^d)$. Then there exists a perfect matching on elements of X with crossing number $O(n^{1-1/d})$ with respect to $\mathcal{S}$.*

Now we are ready to state the main theorem of this section regarding a graph class with a polynomial-sized vertex-separating path system.

Theorem 2.7. *Let $t > 0$ be an integer and G be a $K_{2,t}$-minor-free graph. Let $v_0 \in V(G)$ be a vertex of degree $\Omega(n^\delta)$. Then there exists an $\epsilon := \epsilon(t,\delta) = \frac{\delta}{t+1}$ such that for any vertex-separating path system $\mathcal{P}$ of G, we have $|\mathcal{P}| = \Omega(n^\epsilon)$.*

Proof. Let $\mathcal{P}$ be a separating path system of G. Define the set $\mathcal{P}'$ using $\mathcal{P}$ as follows. For each path $\Pi \in \mathcal{P}$, if $v_0 \in V(\Pi)$ then we include in $\mathcal{P}'$ at most two subpaths of Π created by removing v_0 from Π; otherwise, we include Π in $\mathcal{P}'$. Note that $|\mathcal{P}'| \leq 2 \cdot |\mathcal{P}|$.

Consider the set system $(X, \mathcal{S})$ where $X := N(v_0)$ and $\mathcal{S} := \{X \cap \Pi \mid \Pi \in \mathcal{P}'\}$. Let $(\mathcal{S}, \mathcal{X}^*)$ be the dual set system to $(X, \mathcal{S})$. We claim that the set system $(\mathcal{S}, \mathcal{X}^*)$ has a VC-dimension at most $t+1$. Assume to the contrary that $(\mathcal{S}, \mathcal{X}^*)$ has a VC-dimension of at least $t+2$. Let $\mathcal{Q} := \{\Pi_1, \Pi_2, \ldots, \Pi_{t+1}, \Pi_{t+2}\}$ be a set of $t+2$ distinct paths in $\mathcal{S}$ which are shattered by $\mathcal{X}^*$. By definition, for each $1 \leq i \leq t$, there are distinct vertices $\{x_1, x_2, \ldots, x_t, y_1, y_2, \ldots t_t\}$ such that $x_i \in \Pi_{t+1} \cap \Pi_i$ and x_i is not contained in any other paths in $\mathcal{Q}$. Similarly, $y_i \in \Pi_{t+2} \cap \Pi_i$ and y_i is not contained in any other path in $\mathcal{Q}$. For each y_i, let y_i' be the neighbor of y_i on the path Π_i we meet by traversing it from x_i to y_i. Let Π_i' be the sub-path of Π_i from x_i to y_i'. Let H be the graph obtained by contracting Π_{t+1} and all of the Π_i' to a vertex. By deleting some further edges from H we obtain a star with the leaves $y_1, y_2, \ldots, y_t$. Hence G contains a minor of $K_{2,t}$, a contradiction.

By Lemma 2.3 and Theorem 2.6, there is a matching M of elements in X such that each set $B \in \mathcal{S}$ crosses and therefore separates $O(n^{\delta(1-1/(t+1))})$ pairs of elements in M. Hence to separate every pair in X, we need $\Omega(n^{\delta}/n^{\delta(1-1/(t+1))}) = \Omega(n^{\delta/(t+1)})$ paths in $\mathcal{P}'$. This implies that any vertex-separating path system of G must have size $\Omega(n^{\delta/(t+1)})$. ☐

Remark 2.1. *The dual of a set system with a bounded VC-dimension has a bounded VC-dimension [6]. Using this fact and Lemma 2.3, for a set system $(X, \mathcal{S})$ of VC-dimension d, we have that $\pi^*_{\mathcal{S}}(k) = O(k^{2^{d+1}})$. Therefore, in the proof of Theorem 2.7, to obtain an upper bound for the VC-dimension of the dual set system $(\mathcal{S}, \mathcal{X}^*)$, we might only rely on the upper bound of the VC-dimension of the primal set system $(X, \mathcal{S})$. It is not hard to see that the VC-dimension of the primal set system is also at most $t+1$. However, to obtain an improved lower bound for the size of the vertex-separating path system of G, we directly study the VC-dimension of the dual set system.*

As a corollary of this result, every outerplanar graph with a high degree (polynomial-sized) vertex requires a polynomial-sized vertex-separating path system. On another note, the result of Theorem 2.7 can be generalized to $K_{3,t}$-minor-free graphs in a natural way. In fact, for a positive integer t, if G is a $K_{3,t}$-minor-free graph with two vertices u_0 and v_0, and the common neighborhood of u_0 and v_0 has a size of $\Omega(n^{\delta})$, then there exists an $\epsilon := \epsilon(t, \delta)$ such that any vertex-separating path system of G is required to have a size $\Omega(n^{\epsilon})$. This, in turn, implies that every bounded genus graph with two vertices having a polynomial-sized common neighborhood requires a vertex-separating path system of polynomial size [20, 57, 58].

2.8 Separating Tree Systems

In this section, as a generalization of separating path systems, we study separating tree systems for the vertex set of a graph G. Recall that $f_t(G)$ is the size of a smallest vertex separating tree system of the graph G, that is, the smallest number of subtrees, $T_1, T_2, T_3, \ldots, T_k$, of G, such that each pair of vertices is separated by one of these trees.

As a comparison, notice that in contrast to separating path systems, $f_t(K_{m,n}) = O(\log(n+m))$ [59].

Theorem 2.8. *Let T be an n-vertex tree. Then $\log(n) \le f_t(T) \le n/2 + \log(n) + O(1)$, and these bounds are tight (up to lower order terms).*

Proof. Let v^* be a centroid of T, that is, a vertex of T, such that its removal results in connected components, each of size at most $\frac{n}{2}$. We root each component of $T \setminus \{v^*\}$ at a neighbor of v^*. First, we separate the vertices of different components from each other. With an idea similar to Theorem 2.1, we assign a binary string of length $\lceil \log(deg(v^*)) \rceil \le \log n + 1$ to each component of $T \setminus \{v^*\}$ and for each bit position we consider a spanning tree of v^* and all components whose bit is 1 at this position.

Within each component of $T \setminus \{v^*\}$, we consider the paths from the root of the component to each of its vertices. Note that by the property of centroid, there are at most $\frac{n}{2}$ such paths in each component. We order these paths within each component arbitrarily. Using the vertex v^* as a common neighbor, we merge the paths with the same index into one tree. This set of $\frac{n}{2}$ trees will separate the vertices located in the same component. To separate v^* from all the other vertices, we add one single-vertex tree, covering v^*. In total we have at most $\frac{n}{2} + \log n + O(1)$ trees.

The $\frac{n}{2}$ term in the upper bound is the best possible, because if the tree T is a path, then a vertex-separating tree system is equivalent to a vertex-separating path system and, by Observation 2.1, we know that $f_t(P_n) = \lceil \frac{n}{2} \rceil$. By Theorem 2.1, the lower bound is trivial, and observe that by the above construction, one can see that $f_t(K_{1,n-1}) = \log n + O(1)$. □

In the rest of this section, we prove an upper bound on $f_t(T)$ in terms of the number of vertices and the radius of the tree T. For a non-empty subset S of non-negative integers, define $b(S)$ to be the bitwise OR of the elements of S. If k is the smallest integer such that $\max(S) < 2^k$, define $c(S) := \{2^k - 1 - x | x \in S\}$. In other words, the elements of $c(S)$ are obtained from the elements of S by flipping the first k bits in their binary representation.

Lemma 2.4. *Let $[l_1, r_1]$ and $[l_2, r_2]$ be two disjoint intervals of integers, where, $0 \le l_1 \le r_1 < l_2 \le r_2$. Then $b([l_1, r_1]) \ne b([l_2, r_2])$ or $b(c([l_1, r_1])) \ne b(c([l_2, r_2]))$.*

Proof. Let k be the smallest integer such that $r_2 < 2^k$. We prove the lemma by induction on k. If $k = 1$, we have $l_1 = r_1 = 0$ and $l_2 = r_2 = 1$, therefore, $b([l_1, r_1]) \ne b([l_2, r_2])$. Let $k \ge 2$ and assume that the statement is true for all values less than k. In order to prove the statement for k, we consider three cases:

- $r_1 < 2^{k-1}$: Then the k-th bit in $b([l_1, r_1])$ is 0. While, by definition, the k-th bit in $b([l_2, r_2])$ is 1, we have $b([l_1, r_1]) \ne b([l_2, r_2])$.
- $r_1 \ge 2^{k-1}$ and $l_1 < 2^{k-1}$: Since $l_2 \ge 2^{k-1}$, the k-th bit in $b(c([l_2, r_2])$ is 0. Since $l_1 < 2^{k-1}$, the k-th bit in $b(c([l_1, r_1])$ is 1. Thus, $b(c([l_1, r_1])) \ne b(c([l_2, r_2]))$.

- $l_1 \geq 2^{k-1}$: In this case, the k-th bit is 1 in the binary representation of every integer in $[l_1, r_1] \cup [l_2, r_2]$. In consequence, $b([l_i, r_i]) = b([l_i - 2^{k-1}, r_i - 2^{k-1}]) + 2^{k-1}$ and $b(c([l_i, r_i])) = b(c([l_i - 2^{k-1}, r_i - 2^{k-1}]))$ for $i \in \{1, 2\}$. Therefore, we only need to compare the values for the intervals $[l_1 - 2^{k-1}, r_1 - 2^{k-1}]$ and $[l_2 - 2^{k-1}, r_2 - 2^{k-1}]$, which implies the statement by the induction hypothesis.

□

Theorem 2.9. *Let T be an n-vertex tree with radius r. Then $max(r, \log n) \leq f_t(T) \leq r + 2\lceil \log n \rceil + 1$.*

Proof. We start with proving the lower bound. Let $\mathcal{T}$ be a vertex-separating tree system of T. Let D be a longest path in T; note that $2r - 1 \leq |V(D)|$. The intersection of each element of $\mathcal{T}$ with D is a path. We build a vertex-separating path system of D using $\mathcal{T}' := \{D \cap t | t \in \mathcal{T}\}$. Now, using Observation 2.1, we conclude that $r = \lceil \frac{2r-1}{2} \rceil \leq \lceil \frac{|D|}{2} \rceil \leq |\mathcal{T}|$. On the other hand, by Theorem 2.1, we know that $\lceil \log n \rceil \leq |\mathcal{T}|$. Therefore, $max(r, \log n) \leq f_t(T)$.

To prove the upper bound, let c be a center of the tree T. We root the tree T at c. Note that the height of T is equal to r. We denote the set of leaves of the tree T by $\ell(T)$. (If c has degree one, then we do not consider c to be a leaf.) We will construct two sets of trees.

The first set consists of $r + 1$ trees. For $i = 0, \ldots, r$, the tree T_i is the subtree of T consisting of all the vertices at distance at most i from the root.

The second set consists of $2\lceil \log |\ell(T)| \rceil$ trees. In order to define these trees, we label the leaves of $\ell(T)$ according to a post-order traversal of T with the integers $0, 1, \ldots, |\ell(T)| - 1$. For each $1 \leq i \leq \lceil \log |\ell(T)| \rceil$:

1. Let $\Delta_{0,i}$ be the set of leaves in $\ell(T)$ whose labels have a zero in the i-th position of their binary representations.

2. Let $\Gamma_{0,i}$ be the smallest subtree of T that is rooted at the root of T and for which $\ell(\Gamma_{0,i}) = \Delta_{0,i}$.

3. Let $\Delta_{1,i}$ be the set of leaves in $\ell(T)$ whose labels have a one in the i-th position of their binary representations.

4. Let $\Gamma_{1,i}$ be the smallest subtree of T that is rooted at the root of T and for which $\ell(\Gamma_{1,i}) = \Delta_{1,i}$.

We will show that the trees T_i, $0 \leq i \leq r$, and $\Gamma_{j,i}$, $j \in \{0, 1\}$ and $1 \leq i \leq \lceil \log |\ell(T)| \rceil$, form a separating tree system of the tree T. Let u and v be two distinct vertices of T. We number the levels in T from 0 to r, where the root is at level zero. Let i be the level of u and j be the level of v in T. We consider two cases:

- $i \neq j$. We may assume that $i < j$. Then u is in T_i whereas v is not in T_i.
- $i = j$. Let S_u be the set of labels of the leaves in the subtree of u, and similarly define S_v for the vertex v. Note that S_u and S_v are two disjoint intervals of consecutive integers. By Lemma 2.4, $b(S_u) \neq b(S_v)$ or $b(c(S_u)) \neq b(c(S_v))$. For any of these two inequalities, one of the bit positions in which they differ introduces the tree that separates u from v.

The total number of elements used in the two sets of trees is at most $r+2\lceil \log |\ell(T)|\rceil+1$, which is at most $r+2\lceil \log n\rceil+1$. □

This result extends to all connected graphs. For a graph G of radius r, we can consider a BFS tree rooted at a center of G and use this spanning tree to find a separating system for G.

2.9 Open Problems

The following computational complexity question is still open: Is there a polynomial time algorithm to compute $f(T)$, when T is a tree? However, we conjecture that determining the exact value of $f(G)$ for an arbitrary graph G is NP-complete. Additionally, we conjecture the following:

Conjecture 2.1. *The problem of determining if $f(G) = \lceil \log |V(G)| \rceil$ for any given graph G is NP-complete.*

After studying the grid graphs in Section 2.5, Theorem 2.4 implies the following simple corollary.

Corollary 2.2. *Let G_1 and G_2 be two graphs. If G_1 and G_2 contain Hamiltonian paths, then $f(G_1 \square G_2) \leq O(\log |V(G_1)| + \log |V(G_2)|)$* [8].

In order to generalize this corollary, a natural question arises regarding the relationship of $f(G_1)$ and $f(G_2)$ for graphs G_1 and G_2, respectively, with their products (Cartesian product, strong product[9] , etc.). For instance, in line with Corollary 2.2 we can ask the same question for general graphs G_1 and G_2.

Problem 2.1. *Let G_1 and G_2 be connected graphs. What can we say about $f(G_1 \square G_2)$ as a function of $f(G_1)$ and $f(G_2)$?*

[8] For two graphs G_1 and G_2, the *Cartesian graph product* of G_1 and G_2, denoted $G_1 \square G_2$, is a graph whose vertex set is $V(G_1 \square G_2) := V(G_1) \times V(G_2)$ and that contains an edge between distinct vertices $v = (v_1, v_2)$ and $w = (w_1, w_2)$ if and only if (i) $v_1 = w_1$ and $v_2 w_2 \in E(G_2)$; or (ii) $v_2 = w_2$ and $v_1 w_1 \in E(G_1)$.

[9] For two graphs G_1 and G_2, the *strong graph product* of G_1 and G_2, denoted $G_1 \boxtimes G_2$, is a graph whose vertex set is $V(G_1 \boxtimes G_2) := V(G_1) \times V(G_2)$ and that contains an edge between distinct vertices $v = (v_1, v_2)$ and $w = (w_1, w_2)$ if and only if (i) $v_1 = w_1$ and $v_2 w_2 \in E(G_2)$; or (ii) $v_2 = w_2$ and $v_1 w_1 \in E(G_1)$; or (iii) $v_1 w_1 \in E(G_1)$ and $v_2 w_2 \in E(G_2)$.

A question that could establish a connection between vertex-separating path systems and edge-separating path systems is as follows.

Problem 2.2. *Is there a relation between $f(G)$ and $f(\mathcal{L}(G))$, where $\mathcal{L}(G)$ is the line-graph of G?*

After exploring the maximal outerplanar graph in Section 2.6, the next family of graphs to examine is maximal planar graphs. It is worth noting that some maximal planar graphs have a vertex-separating path system with linear size. However, determining the precise upper bound for this class of graphs is the next question we would like to ask.

Problem 2.3. *What is the value of $f(\Delta_n)$, where Δ_n is an n-vertex triangulation?*

As a final note, it would be worth exploring a tight upper bound for the size of vertex-separating tree systems on different graph classes, such as maximal outerplanar graphs, and triangulations.

Acknowledgement

This research was initiated and conducted at the Tenth Annual Workshop on Geometry and Graphs, held at the Bellairs Research Institute in Barbados, February 3 – February 7, 2023. The authors are grateful to the organizers and to the participants of this workshop. We would like to thank the anonymous reviewers for careful reading of our manuscript.

Chapter 3

Euclidean Freeze-Tag Problem on the Plane

Nicolas Bonichon[1] Cyril Gavoille[1] Nicolas Hanusse[1] Saeed Odak[2]

Abstract. The freeze-tag problem is an optimization problem introduced by Arkin et al. (SODA'02). This problem revolves around efficiently waking up a swarm of inactive robots starting with a single active robot. Each inactive robot is awakened by an active robot going to its location. The objective is to minimize the total wake-up time for all robots, the *makespan*.

A recent paper by Bonichon et al. [16] considers the geometric version of the freeze-tag problem on the plane. They conjectured that for the robots located on the plane with ℓ_2-norm, the makespan is at most $(1 + 2\sqrt{2})r$, where r is the maximum distance between the initial active robot and any inactive robot (Assume the robots move at a speed of 1 unit per second). In this chapter, we prove the conjecture for the robots in convex position and for $n \leq 7$ and $n \geq 281$, where n is the number of inactive robots (The conjecture was known to be true for $n \geq 528$ robots as shown by Bonichon et al. [16]). Moreover, we show an upper bound of $4.63r$ for the makespan of robots in a disk of radius r in the ℓ_2-norm, improving the best known bound of $5\sqrt{2}r \approx 7.07r$.

3.1 Introduction

The *freeze-tag problem* is an optimization problem concerned with waking up a swarm of inactive robots in the shortest possible time starting with a single active robot. Consider a set of robots represented by S and $|S| = n + 1$ for $n \in \mathbb{N}$. Let $p_0, \ldots, p_n$ be the locations of the robots in a metric space, with p_0 being the location of the initial active robot. Each inactive robot is awakened (activated) by an active robot going to its location.[3] As soon as an inactive robot is activated, it can assist in waking up the other

[1]LaBRI, University of Bordeaux, CNRS, Bordeaux INP, France
[2]School of Electrical Engineering and Computer Science, University of Ottawa, Canada.
[3]We use the terms "waking up" and "activating" a robot interchangeably throughout this chapter.

inactive robots. We assume that each active robot moves at the same speed of one unit per second while the inactive robots do not move. The *makespan* (*wake-up time*) is the time of the last wake-up. The objective, in the freeze tag problem, is to minimize the makespan. The freeze-tag problem has applications in group formation, searching, and recruitment in robotics, as well as broadcasting and IP multicast problems in network design (see [3, 50] and their references).

The problem can be rephrased as follows: A *wake-up tree* of S is a binary weighted spanning tree rooted at p_0 such that the degree of p_0 is one and the length of an edge is the distance between its endpoints (see for instance Figure 3.1). The freeze-tag problem is to find a wake-up tree of S with the minimum (weighted) height, where height is the maximum weighted path length (sum of edge weights) from the root to any leaf vertex.

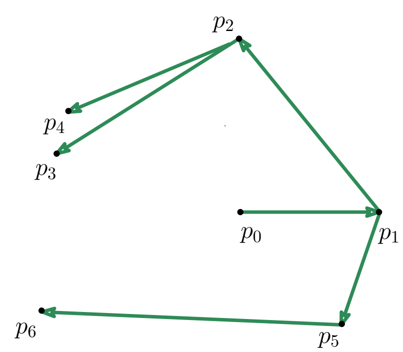


Fig. 3.1: An example of a wake-up tree with 6 inactive robots in the Euclidean plane.

Note that the possible movements of robots in the freeze-tag problem can be modeled with a complete weighted graph with vertices corresponding to the positions of the robots, and the weight of each edge is equal to the distance between two corresponding robots in the metric space. In this setup, Arkin et al. [4] give a constant approximation algorithm for the freeze-tag problem, when one inactive robot is located on each vertex. They also show that using an underlying graph metric, the problem is NP-hard. In a different paper, Arkin et al. [3] show that obtaining a 5/3-approximation is NP-hard for general metrics on weighted graphs. Therefore, a polynomial-time approximation scheme does not exist unless $P = NP$. In a related paper, Könemann et al. [50] consider the problem of finding a minimum diameter spanning tree with a bounded maximum degree in a complete undirected weighted graph and provide an $O(\sqrt{\log n})$-approximation algorithm for the freeze-tag problem in the general setting.

In this chapter, we consider the *geometric freeze-tag problem* for the collection of robots. In the geometric freeze-tag problem, robots are modeled as points in $\mathbb{R}^d$ in a particular metric for some $d \in \mathbb{N}$. For $d = 3$ and ℓ_p norm, it has been shown that the geometric freeze-tag problem is NP-hard where $p \geq 1$ [24, 31, 55]. Sztainberg et al. [63] give a heuristic algorithm with a tight approximation of $\Theta(\log^{1-1/d} n)$ for the makespan of n inactive robots in d dimensional space. In particular, their greedy algorithm yields an $O(1)$-approximation in one dimension ($d = 1$) and an $O(\sqrt{\log n})$-approximation in two dimensions ($d = 2$). Arkin et al. in [3], for any constant $d \in \mathbb{N}$, provide a polynomial-time approximation scheme when robots are located in $\mathbb{R}^d$ equipped with ℓ_p metric. Moreover, their algorithm runs in time $O(n \log n + 2^{\text{poly}(1/\epsilon)})$.

It is worth mentioning that Hammar et al. [41] study the online freeze-tag where each inactive robot is revealed at a specified time. Later, an optimal algorithm for the online freeze-tag problem was introduced by Burnner et al. [21].

In the geometric setting, as long as normed spaces are concerned, the positions of all the robots can be scaled and translated such that all inactive robots are in a unit disk, and the initial active robot is at the origin (i.e., the active robot is at the origin and the distance between the active robot and the farthest inactive robot is a unit). Note that in this configuration, the makespan is always lower bounded by the maximal distance between the active robot and inactive robots (the radius of the unit disk). Combinatorial upper bounds for the makespan of robots in a unit disk with respect to ℓ_p norm are studied by Bonichon et al. [16]. In particular, when robots are located in the unit disk in the plane with ℓ_1 norm, they provide a tight strategy with makespan 5. They also show [16, Proposition 15] that the makespan for n inactive robots in the unit disk with one active robot at the origin is at most $3 + c/\sqrt{n}$, where c is a constant depending on the norm.

We focus on the *Euclidean freeze-tag problem on the plane*. That is, we consider robots as points in the Euclidean metric space on $\mathbb{R}^2$. In this setting, the problem remains NP-hard [71], and Najafi Yazdi et al. [70] provide an algorithm with a makespan $(5+2\sqrt{2}+\sqrt{5})$ for the robots located in a unit square that runs in linear time. Recently, Bonichon et al. [16] proposed an algorithm with a makespan $5\sqrt{2}$. They also conjectured that the maximum makespan of robots in a unit disk of any norm is achieved when the number of robots is four. For $n = 4$, we get the worst-case whenever four inactive robots p_1, p_2, p_3, and p_4 form a square with sides of length $\sqrt{2}$. It takes time 1 to go from the active robot p_0 in the center to p_1, and then one robot has to wake up p_2 followed by p_3 in time $2\sqrt{2}$, and the other one wakes up p_4. In the Euclidean freeze-tag problem, this translates to the following conjecture.

Conjecture 3.1 ([16])**.** *Let n be a positive integer. There exists a strategy to wake up n inactive robots inside a unit disk in Euclidean space starting with an active robot at the origin in time at most* $1 + 2\sqrt{2} \approx 3.83$.

Our main contributions are the following. First we show that Conjecture 3.1 holds for $n \leq 7$ and $n \geq 281$, and also when the robots are in convex position (Theorems 3.1, 3.2, and 3.3). Then we provide a new upper bound of 4.63 for the makespan of the Euclidean freeze-tag problem on the plane, improving upon the best-known result of $5\sqrt{2} \approx 7.07$. This also shows that the optimal upper bound for the Euclidean case is strictly less than the lower bound of 5 for the ℓ_1 norm.

The rest of this chapter is organized as follows: The next section will be dedicated to preliminaries and some definitions for the geometric objects needed in the sequel. In Section 3.3, we discuss monotonic wake-up strategies for two simple geometric objects

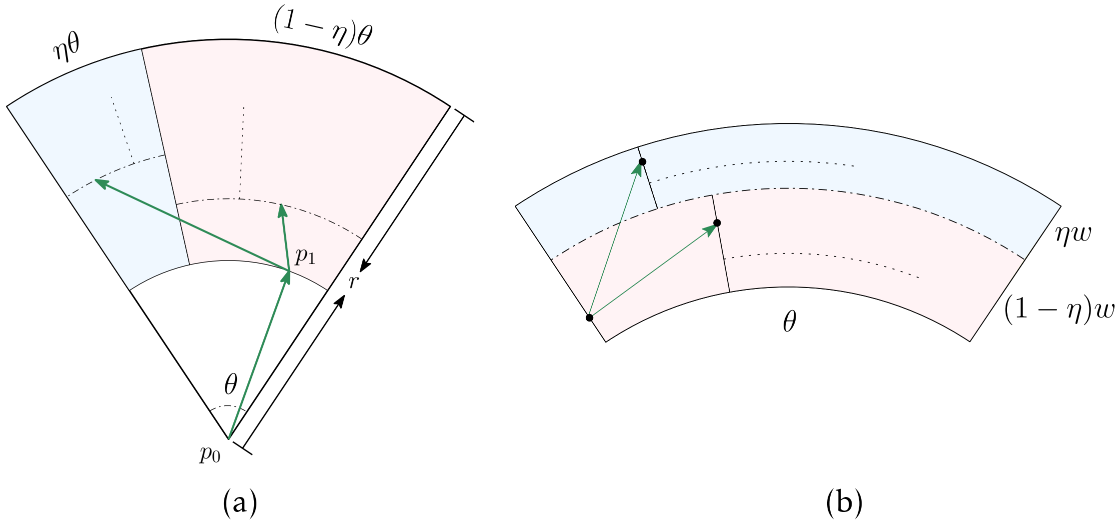


Fig. 3.2: (a) A cone of angle θ and radius r. The monotonic wake-up strategy to solve a cone with one active robot in the apex. (b) A crown of angle θ and width w. The monotonic wake-up strategy to solve a crown with two active robots on a corner.

as a subroutine. In Section 3.4, as a warm-up, we prove Conjecture 3.1 for small values of n. Section 3.5 and Section 3.6 study the correctness of Conjecture 3.1 when the inactive robots are in a convex position and when the number of inactive robots is at least 281, respectively. Section 3.7 establishes an improved makespan of 4.63 for the Euclidean freeze-tag problem in the plane.

3.2 Preliminaries

For each $0 \leq i \leq n$, the *wake-up time of robot* p_i is the length of the path from p_0 to p_i in the wake-up tree. The height of a wake-up tree indicates its makespan. Using this terminology, a closed geometric region $\mathcal{R}$ on the plane containing a specified active robot has a makespan of at most τ if for every $n \in \mathbb{N}$, there exists a wake-up tree for every configuration of n inactive robots in the region $\mathcal{R}$ with a height at most τ.

Many of the strategies that we will define rely on a recursive decomposition of a region $\mathcal{R}$ into subregions. Therefore, we will define some regions that will be useful to us later on.

A *cone* of angle θ and radius r is a geometric region inside a disk of radius r between two segments with one endpoint on the center of the disk and the other endpoint on the boundary such that the angle between the two segments is θ (see Figure 3.2(a)). The center of the disk is referred to as the cone's apex. A cone defined using a disk of radius one is called a *unit cone*.

A (unit) *crown* of angle θ and width w is obtained from a unit cone of angle θ by subtracting a smaller cone of the same angle and radius $1 - w$ (see Figure 3.2(b)). Each non-trivial crown consists of 4 sides: two curved sides and two straight-line sides. We call the longer curved side of a crown the *exterior side* and the shorter curved side of a crown is called the *interior side*. For future reference, we represent the makespan of a unit crown of angle θ and width w starting with one active robot at a corner on the interior side of the crown with crown(w, θ).

Throughout this chapter, ϕ stands for the golden ratio, i.e., $\phi = \frac{1+\sqrt{5}}{2}$. Note that $\phi^2 = \phi + 1$.

3.3 First Bounds For Geometric Shapes

We begin this section with a simple observation stating the triangle inequality in polar notation.

Observation 3.1. *Let $A = (r_a, \theta_a)$ and $B = (r_b, \theta_b)$ be two points in polar notation inside a unit disk. Then we have $\|AB\| \le |r_a - r_b| + \max(r_a, r_b) \cdot |\theta_a - \theta_b|$. In particular, since $r_a, r_b \le 1$, we have $\|AB\| \le |r_a - r_b| + |\theta_a - \theta_b|$.*

In the following, $|r_a - r_b|$ and $\max(r_a, r_b) \cdot |\theta_a - \theta_b|$ are referred to as the radial distance and angular distance between A and B, respectively. Given a binary wake-up tree, a strategy is considered *monotonic* if, for every path from the root to the leaves, the points are ordered by their distance from the root.

In this section, we present monotonic wake-up strategies for two simpler geometric objects, namely the unit cone and the unit crown, as subroutines for the other algorithms discussed in the rest of the chapter. We first present a result from [16] that establishes an upper bound for the makespan of robots positioned within a unit cone of angle θ (refer to Figure 3.2(a)).

Monotonic Cone Strategy

We describe the monotonic wake-up tree strategy mentioned in Bonichon et al. [16] for a cone. We start with one active robot, p_0, in the apex of the cone. In the first step, the active robot finds the closest robot to the origin, say p_1, and activates it. Next, they divide the remaining region (a crown) into two parts with ratio $\eta : (1-\eta)$ (the constant η is a real number between zero and one, to be determined later). That is, the remaining region is partitioned into two crowns with angles $\eta\theta$ and $(1-\eta)\theta$ such that the crown with angle $(1-\eta)\theta$ contains the robot p_1 on the boundary (see Figure 3.2(a)). Next, p_0 is assigned to wake up the rest of the robots in the crown of angle $\eta\theta$, and p_1 is assigned to the crown of angle $(1-\eta)\theta$. The robots p_0 and p_1 follow similar strategies within their respective crowns. Each of them finds the closest robot to the origin inside their crown and wakes it up. Next, by the new active robot, they repeat in smaller regions by dividing it according to the ratio $\eta : (1-\eta)$ as before and then proceed similarly.

Lemma 3.1 ([16], Proposition 14)**.** *The **Monotonic Cone Strategy** wakes up inactive robots in a cone of angle θ and radius one starting with one active robot at the apex of the cone in time at most $1 + \phi\theta$.*

Proof. Using Observation 3.1, the length of each segment in the total movement of each robot is upper-bounded by the sum of its radial distance and angular distance. Since the movements are monotonic along the radius, the total radial distance of each robot is at most 1, the radius of the cone. Therefore, we consider an upper bound for the total angular movements of a robot. Note that the first step of the algorithm does not have an angular movement. Let $A(\theta)$ represent an upper bound on the total angular movement of a robot in the wake-up tree of a crown of angle θ starting with two active robots on the interior side. By considering a worst-case scenario, as an upper bound for the value of $A(\theta)$, we have the following:

$$A(\theta) \leq \max\{\theta + A(\eta\theta),\ (1-\eta)\theta + A((1-\eta)\theta)\}.$$

Recall that we split a crown of angle θ into two crowns of angle $\eta\theta$ and $(1-\eta)\theta$, while the two active robots are located on the boundary of the crown with angle $(1-\eta)\theta$. This inequality arises from the scenario where one of the active robots might need to travel to the opposite side, resulting in an angular distance of at most θ. Meanwhile, the other robot might need to travel from one side to the dividing line, resulting in an angular distance of at most $(1-\eta)\theta$. By definition of $A(\theta)$, after activating the next robot in each region, the crowns will have angular movement of $A(\eta\theta)$ and $A((1-\eta)\theta)$, respectively. We have to consider a value of η that balances the height of the two branches of the wake-up tree at the position of the newly awakened robot. Note that $A(\theta)$ is an increasing function with respect to θ. Therefore, $\theta + A(\eta\theta)$ is increasing and $(1-\eta)\theta + A((1-\eta)\theta)$ is decreasing with respect to θ. To expect the smallest value on the evaluation of the upper bound for $A(\theta)$, we set the value of η such that $\theta + A(\eta\theta)$ and $(1-\eta)\theta + A((1-\eta)\theta)$ are equal. By setting $\eta = 2-\phi$, we have $A(\theta) \leq \sum_{i=0}^{\infty} \eta^i \theta \leq \phi\theta$. Therefore, the total makespan is upper bounded by $1 + A(\theta) \leq 1 + \phi\theta$. □

Monotonic Crown Strategy

As the next geometric subroutine, similar to **Monotonic Cone Strategy** (Lemma 3.1), we can construct a monotonic wake-up strategy for a unit crown using a monotonic recursive partition into sub-crowns.

We start with two active robots at a point on the left boundary of the crown as depicted in Figure 3.2(b). Note that two robots can be on the corner. We use the monotonic wake-up algorithm. This time, we split the crown into two parallel crowns. The width of the original crown is divided into two parts with a ratio $\eta : (1-\eta)$ where the part with the coefficient $1-\eta$ contains both of the active robots (η is a constant between zero and one to be determined later). Without loss of generality, we assume that each point of the dividing curve has distance $1-\eta w$ from the center (refer to Figure 3.2(b)).

Each of the active robots takes care of a different crown to wake up. The first robot finds, in the angular distance, the closest inactive robot in the crown with width $(1-\eta)w$ and angle θ, and the second robot finds, in the angular distance, the closest inactive robot in the crown with width ηw and angle θ (see Figure 3.2(b)). When each of the robots wakes up another robot, they split the corresponding region similarly with ratio $\eta : (1-\eta)$ and they repeat in the smaller regions to wake up the remaining robots.

Lemma 3.2. *The* **Monotonic Crown Strategy** *wakes up all of the robots in a crown of angle θ and width w starting with two active robots at a corner in time at most $\theta + \phi w$.*

Proof. We use Observation 3.1 and worst-case analysis to provide an upper bound for the makespan of this algorithm. Note that the angular movement of the robots with respect to the origin is increasing. Therefore, using Observation 3.1, we upper bound the entire angular movement of any robot with θ, the length of the exterior side of the crown.

Denote by $\mathrm{R}(w)$ the total radial movement spent by a robot to wake up all the other robots in a crown of width w. To calculate the radial movement of each robot, we apply a worst-case analysis. By examining the two main branches in the first step of the wake-up tree, we obtain the following recurrence.

$$\mathrm{R}(w) \leq \max\{w + \mathrm{R}(\eta w), (1-\eta)w + \mathrm{R}((1-\eta)w)\}.$$

This formula is analogous to the one in the analysis of **Monotonic Cone Strategy** (Lemma 3.1). Note that $\mathrm{R}(w)$ is an increasing function with respect to the w. Therefore, $w + \mathrm{R}(\eta w)$ is increasing and $(1-\eta)w + \mathrm{R}((1-\eta)w)$ is decreasing with respect to w. By setting $\eta = 2 - \phi$, we have that $w + \mathrm{R}(\eta w) = (1-\eta)w + \mathrm{R}((1-\eta)w)$ and $\mathrm{R}(w) \leq \sum_{i=0}^{\infty} \eta^i w \leq \phi w$. Therefore, the total makespan is upper bounded by $\theta + \mathrm{R}(w) \leq \theta + \phi w$. □

Note that in **Monotonic Crown Strategy**, we started with two active robots on the boundary of the crown. If we consider only one active robot on the boundary, we must consider an extra time to wake up another robot, and then we can apply the result of **Monotonic Crown Strategy**. By a simple analysis of the first step to waking up the second robot inside the crown with an angular sweep, we have the following corollary.

Corollary 3.1. *There exists a strategy to wake up all of the robots in a crown of angle θ and width w starting with one active robot at a corner in time at most $\theta + (1+\phi)w$.*

Finally, with a strategy analogous to that in **Monotonic Crown Strategy**, one can wake up robots within a rectangular region.

Monotonic Rectangle Strategy

Let inactive robots be located in a rectangle with width w and height h. Assume we start with two active robots at a point on the left boundary of the rectangle. We split the rectangle into two parallel rectangles. We divide the rectangle into two smaller rectangles by dividing its height with a ratio $\eta : (1-\eta)$ where the part with the coefficient $1-\eta$ contains both of the active robots (again $\eta = 2 - \phi$). Each of the active robots takes care of a different rectangle to wake up. The first robot finds, with a horizontal sweep, the closest inactive robot in the rectangle with height $(1-\eta)h$, and the second robot finds, the closest inactive robot in the rectangle with height ηh. When each of the robots wakes up another robot, they split the corresponding region similarly with ratio $\eta : (1-\eta)$ along the height and they repeat in the smaller regions to wake up the remaining robots. Now, we can state the following corollary, with an exact analysis as in Lemma 3.1 and Lemma 3.2.

Corollary 3.2. *The **Monotonic Rectangle Strategy** wakes up all of the robots in a rectangle of width w and height h starting with two active robots at a corner in time at most $w + \phi h$.*

3.4 Configurations With Small Number of Inactive Robots

In this section, we show the correctness of Conjecture 3.1 for the small number of inactive robots ($n \leq 7$). Note that the first active robot p_0 is located on the center of the unit disk.

Small Group Strategy

For $n \leq 3$, the robot p_0 wakes up the closest inactive robot and then these two robots wake up the (at most) two remaining inactive robots in time at most $3 < 1 + 2\sqrt{2}$. Let $n \in \{4,5,6,7\}$ and p_1 be one of the robots such that the line passing through p_0 and p_1 cuts the unit disk C into two half disks, each of which contains at most $\lceil (n-1)/2 \rceil$ robots (different from p_0 and p_1). Let A and A' be the intersection of this line with C, such that p_1 is on the segment p_0A.

The wake-up strategy is as follows: once the robot p_1 is awakened, the two robots are positioned at point A. Then, each robot takes care of half of the disk. In each half-disk, the robot will awaken the robot closest to A, and then the two robots will each awaken at most one more robot.

Theorem 3.1. *Let $n \leq 7$ be a positive integer. The **Small Group Strategy** wakes up n robots in a unit disk in time less than $1 + 2\sqrt{2}$.*

Proof. Let us analyze the height of the wake-up tree in the **Small Group Strategy** for $4 \leq n \leq 7$. Without loss of generality, assume that the segment AA' is horizontal and that A is to the right (see Figure 3.3). Let p_2 be the closest robot to the point A in the upper half-disk. Let B (resp. B') be the intersection of the circle centered at A with radius $\|Ap_2\|$ with the upper half of C (resp. the segment $A'A$). Let C' be the region of the upper half-disk excluding the disk centered at A with radius $\|Ap_2\|$.

Let us first show that the diameter of this region is equal to $\|A'B\|$. The region C' is contained in the convex region defined by the points A', B, B'. In a convex region, a necessary condition for the segment XY to have a length equal to the diameter is that each endpoint (say X) is such that X is in a corner of the region or that the line (X, Y) is normal to the boundary at X (otherwise we can enlarge the segment XY by

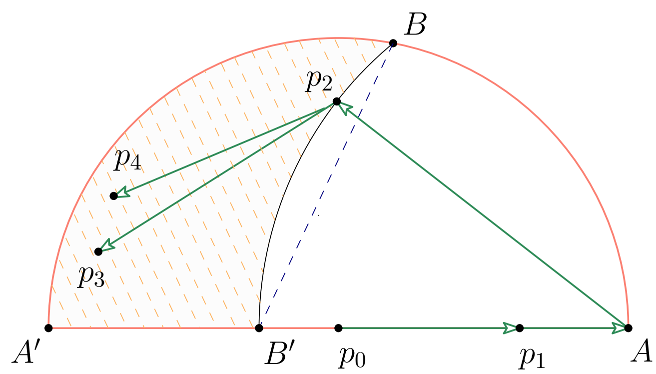


Fig. 3.3: Wake-up strategy with at most 7 robots.

moving X slightly on the boundary). The only segments satisfying this property are the segments $A'B$, $A'B'$, and $B'B$. Therefore, the diameter of this region is equal to $\max(\|A'B\|, \|A'B'\|, \|B'B'\|)$. Furthermore, since $\angle A'B'B > \frac{\pi}{2}$, this diameter is equal to $\|A'B\|$.

Consider a root-to-leaf path p_0, p_1, p_2, p_3 of the wake-up tree. The vertex p_2 is awakened at time $1 + \|AB\|$. Moreover, since the edge p_2p_3 is within the region C', its length is less than the diameter of this region, $\|BA'\|$.

Therefore, the length of the path is at most $1 + \|AB\| + \|BA'\|$. Since B is a point on the half-circle with diameter AA', $\|AB\| + \|BA'\| \leq 2\sqrt{2}$. □

3.5 Robots In Convex Position

In this section, without loss of generality, we assume that the coordinates $p_1, p_2, p_3, \ldots, p_n$ are ordered in the counter-clockwise cyclic ordering around the initial active robot p_0 (center of the the disk). We show the following theorem:

Theorem 3.2. *If the point set corresponding to S (set of sleep robots) is in convex position within a disk C of radius one, the makespan of S is upper bounded by* $1 + 2\sqrt{2}$.

Proof. Let us outline the steps to construct a wake-up tree for inactive robots in convex position.

- For each point p_i of S, we assign a point p_i' on the disk C such that for any pair p_i and p_j, $\|p_i'p_j'\| \geq \|p_ip_j\|$ (see Lemma 3.4).
- If S' is a point set on the disk C, we provide a wake-up tree T' of makespan less or equal to $1 + 2\sqrt{2}$.
- The wake-up tree T on S is defined from T'. If (p_i', p_j') belongs to T' then (p_i, p_j) belongs to T. From Lemma 3.4, the makespan of T is smaller or equal to the one of T' and thus bounded by $1 + 2\sqrt{2}$.

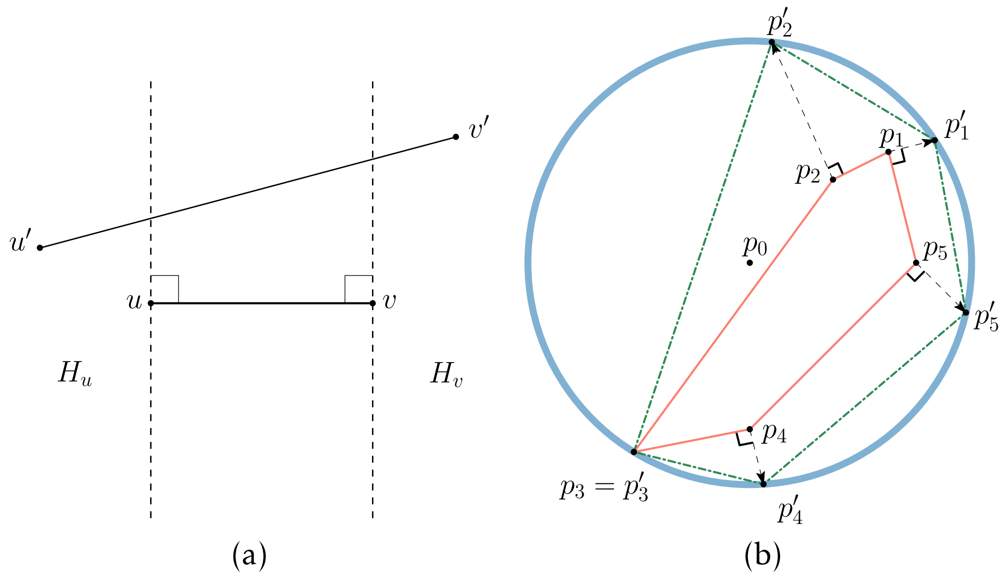


Fig. 3.4: (a) Lemma 3.3 shows how to assign u' (resp. v') to u (resp. v) so that $\|u'v'\| \geq \|uv\|$; (b) Projection of convex points on the disk.

To analyze the height of this wake-up tree, we start with a simple lemma:

Lemma 3.3. *Let u, v be two points. Let H_u (resp. H_v) be the half-space bounded by the perpendicular line L_u (resp. L_v) to the segment uv going through u (resp. v) that does not contain v (resp. u). Let $u' \in H_u$ and $v' \in H_v$. We have $\|u'v'\| \geq \|uv\|$.*

Proof. Let $u'' \in L_u$ and $v'' \in L_v$ be the intersection points of the segment u', v' (see Figure 3.4(a)). Since L_u and L_v are parallel, the distance between u'' and v'' is at least $\|uv\|$. Thus $\|u'v'\| \geq \|u''v''\| \geq \|uv\|$. □

We now define a projection g of S on C defined as follows: For each $p_i \in S$, we assign the point $p_i' \in C$ being the intersection of the ray perpendicular to $p_{i-1}p_i$ emanating from p_i and going outside from the convex hull of S.

Lemma 3.4. *For each pair of points $p_i, p_j \in S$, $\|\mathrm{g}(p_i)\mathrm{g}(p_j)\| \geq \|p_i p_j\|$.*

Proof. Let $p_i' = \mathrm{g}(p_i)$ and $p_j' = \mathrm{g}(p_j)$. Let $\alpha = \angle p_j p_i p_{i-1}$. Let A be any point on the perpendicular line to $p_j p_i$ going through p_i such that $\angle p_j p_i A = \frac{\pi}{2}$. Since S is in a convex position, $\alpha \in [0, \pi]$. We have $\angle A p_i p_i' = \angle p_j p_i p_i' - \angle p_j p_i A = \angle p_j p_i p_{i-1} + \angle p_{i-1} p_i p_i' - \angle p_j p_i A = \alpha + \frac{\pi}{2} - \frac{\pi}{2} = \alpha \in [0, \pi]$. Let ℓ be the line passing through A and p_i. Observe that p_i' and p_j are on two different sides of the line ℓ. Similarly, p_j' and p_i are not on the same side of the line parallel to ℓ going through p_j. Using Lemma 3.3, we conclude that $\|p_i' p_j'\| \geq \|p_i, p_j\|$. □

Arc Strategy

Let's introduce a simple strategy to wake up an arc of angle α containing k inactive robots while the first robot p_j on the arc in the counter-clockwise order is awake. In the **Arc Strategy**, p_j wakes up the adjacent robot p_{j+1}, and any monotonic complete binary wake-up tree can be considered to wake-up rest of inactive robots along the arc. By monotonic, we mean that each robot wakes up another robot with a larger angular position. Note that the height of this binary wake-up tree is $1 + \lfloor \log_2 k \rfloor$.

Disk Strategy

Disk Strategy takes as an input the point set S on the boundary of a unit disk and an angle α_0. Take $\beta = \pi - 2\sqrt{2}$ and let $1 \leq i \leq n$ be an integer where the angle $\angle p_i p_0 p_{i+1}$ is the smallest. We consider two cases (see Figure 3.5):

- Case (I): If $\angle p_i p_0 p_{i+1} \leq \alpha_0$ then p_0 wakes up p_i. For convenience, assume that the line passing through p_0 and p_i is horizontal. After waking up the robot p_i, there are two active robots located at p_i (p_0 and p_i). They wake up in parallel rest of the inactive robots at an angular distance at most $\pi - \beta$ from p_i; one going in the counter-clockwise order through p_{i+1} and the other one in the clockwise order through p_{i-1}. As soon as p_{i+1} is active, it directly goes to the position of the disk at an angular distance $\pi - \beta$ from p_i and wakes up all the remaining robots in the remaining arc of angle 2β. The robot that goes in clockwise order wakes up the robots $p_{i-1}, p_{i-2}, p_{i-3}, \ldots$ in a sequential order and stops at the angular distance $\pi - \beta$.

- Case (II): If the previous case does not hold, take an integer $1 \leq j \leq n$ such that the angle $\angle p_{j-1} p_0 p_{j+1}$ is the smallest. Again, for convenience, assume that robot p_0 and p_j are located on a horizontal line. As in the previous case, p_0 wakes up p_j and robots on the two arcs up to the angular distance $\pi - \alpha_0$ from p_j are awake by two robots emanating from p_j using **Arc Strategy**. One robot from p_{j+1} (resp. p_{j-1}) directly goes to the last robot on the arc $(\pi - \alpha_0, \pi)$ (resp. $(\alpha_0 - \pi, -\pi)$). Then the rest of the robots on the arcs of angle $\pi - \alpha_0$ are awake.

Lemma 3.5. *Assume that every point of S is located on the boundary of the unit disk C. **Disk Strategy** with parameter $\alpha_0 = \pi/11$ builds a wake-up tree having a makespan less than or equal to $1 + 2\sqrt{2}$.*

Proof. Let us start with a simple property: given an angle $\alpha \leq \pi$, the sum of the length of two chords of angles α_1 and α_2 such that $\alpha_1 + \alpha_2 = \alpha$ is maximized when α_1 and α_2 are as close as possible, that is $\alpha_1 = \alpha_2 = \alpha/2$. Let us prove this claim for $\alpha \leq \pi$. Assume x be the smallest of two angles α_1 and α_2, take $f(\alpha, x) = \text{chord}(\alpha_1) + \text{chord}(\alpha_2) = 2\sin(x/2) + 2\sin(\alpha/2 - x/2)$. The function f is symmetric with respect to $x = \alpha/2$ and its derivative $f'(x) = \cos(x/2) - \cos(\alpha/2 - x/2) \geq 0$ for $x \leq \alpha/2$. The function f is maximized whenever $x = \alpha/2$. As a generalization, let t be a positive integer and $0 \leq \alpha \leq \pi$ be a

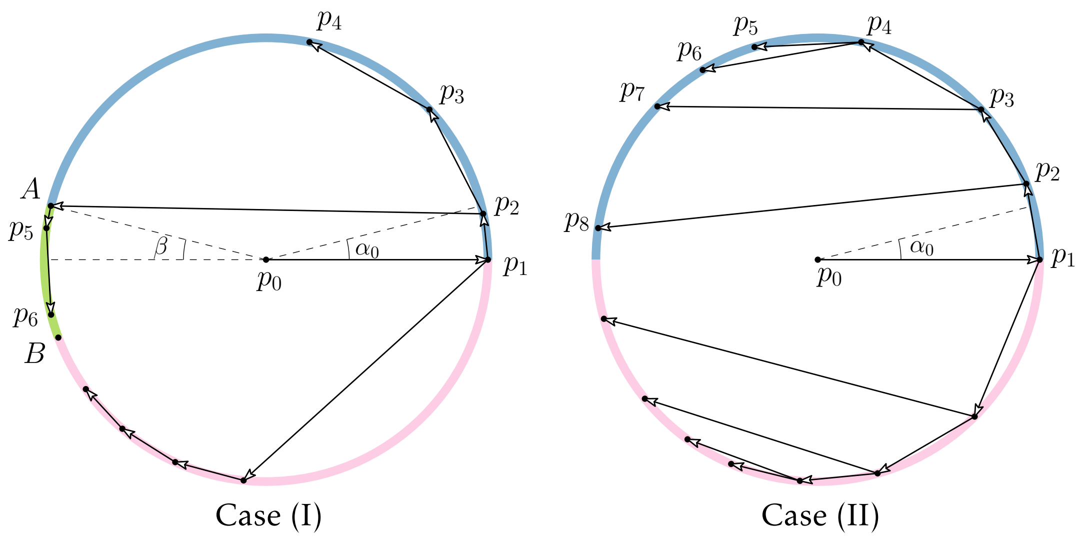


Fig. 3.5: Robots on the disk. Case (I): The disk is partitioned into 3 arcs. Case (II): Without the existence of a small angle, the disk is partitioned into 2 half-disks. Except for the last robot p_8, robots p_2 to p_7 are awake using a monotonic complete binary tree from p_1.

fixed real. Let $\alpha_1, \alpha_2, \ldots, \alpha_t$ be positive reals numbers such that $\sum_{1 \le i \le t} \alpha_i = \alpha$. Note that $\sin x$ for $x \in [0, \pi]$ is a concave function. By Jensen's inequality [46], we have $t \cdot \sin(\alpha/(2t)) \ge \sum_{1 \le i \le t} \sin(\alpha_i/2)$. Thus, we get $\sum_{1 \le i \le t} \text{chord}(\alpha_i) \le 2t \cdot \sin(\alpha/(2t))$.

Assume we are in Case (I), that is $\alpha_0 \le \pi/11$. Let A (resp. B) be the the position at angular distance $2\sqrt{2}$ (resp. $-2\sqrt{2}$) from the robot located at p_i. By construction, every robot located in arcs (p_i, A) and (p_i, B) are awake in time less than $1 + 2\sqrt{2}$ since it takes 1 time unit to go from p_0 to p_i. Let us focus on robots on the arc (A, B). Using the argument in the preceding paragraph, we can upper bound the length of the path p_i, p_{i+1}, A. Since $f(\alpha, x)$ is increasing for $x \le \alpha/2$, this length is upper bounded by $2\sin(\alpha_0/2) + 2\sin((2\sqrt{2} - \alpha_0)/2) < 2.196$. Note that the length of the arc AB is $2\beta = 2 \cdot (\pi - 2\sqrt{2}) < 0.627$. The path p_0, p_i, p_{i+1}, A followed by the arc AB has total length less than $1 + 2.196 + 0.627 = 3.823 < 1 + 2\sqrt{2}$.

Note that Case (I) always holds for $n \ge 22$ since the smallest angle $\angle p_i p_0 p_{i+1} \le 2\pi/n$. Now assume that we are in Case (II) implying that $n < 22$. We know that the smallest angle $\angle p_{j-1} p_0 p_{j+1} \le 4\pi/n$.

Let us focus on a single half-disk. Let x be the angle $\angle p_j p_0 p_{j+1}$. Since we are in case (II), $\pi/11 < x < 4\pi/n - \pi/11$, and we have at most one robot in the arc $(\pi - \alpha_0, \pi)$. Thus, we have two types of root-to-leaf paths in the wake-up tree:

- The path p_0, p_j, p_{j+1}, p_k where p_k is the last robot in the half-disk containing p_{j+1};
- Paths using Arc Strategy for the robots from p_{j+1} to p_{k-1}.

For the path of the first kind, its length is $\|p_0p_j\| + \|p_jp_{j+1}\| + \|p_{j+1}p_k\| = \|p_0p_j\| + \text{chord}(x) + \text{chord}(\angle p_{j+1}p_0p_k) \leq 1 + \text{chord}(x) + \text{chord}(\pi - x) \leq 1 + 4\sin(\pi/4) = 1 + 2\sqrt{2}$.

For the paths of the second kind, we know that the maximal number of robots per half-disk is 11, otherwise we are in Case (I). It means that we have to wake up at most 9 from p_{j+1}. Thus, using Arc Strategy, the wake-up tree rooted at p_{j+1} has height at most $1 + \lfloor \log_2 9 \rfloor = 4$. In the wake-up tree, by Jensen's inequality [46], any monotonic path of 4 hops from p_{j+1} to $p_{j'}$, with $j' < k$ has length at most $4 \cdot \text{chord}((\pi - \alpha_0 - x)/4)$.

Let $h(x) = \text{chord}(x) + 4 \cdot \text{chord}((\pi - \alpha_0 - x)/4)$. Its derivative is $h'(x) = \cos(x/2) - \cos((\pi - \alpha_0)/8 - x/8)$ being strictly positive for $x < (\pi - \alpha_0)/5$ and negative otherwise. Thus $h(x) \leq h((\pi - \alpha_0)/5) = 10 \cdot \sin(\pi/11) < 2.818$. Thus we have that the length of paths of the second kind is less than $1 + 2.818 = 3.818 < 1 + 2\sqrt{2}$. □

This ends the proof of Theorem 3.2. □

3.6 Strategies For Large Number of Robots

Bonichon et. al. [16, Proposition 15] show that for ℓ_2-norm, the time to wake up n inactive robots starting with one active robot in the center of a unit disk is at most $3 + \frac{4\phi\pi}{\lceil 1+\sqrt{1+n}\rceil}$. By examining the constants closely, Proposition 15 in [16] shows that Conjecture 3.1 is correct when n, the number of inactive robots, is at least 528. In line with this result, we state that Conjecture 3.1 holds when the number of inactive robots in the unit disk is at least 281. To get this lower bound on the number of robots, we need to introduce a wake-up strategy for the domes and semi-cones.

A *dome* is the part of a cone between its arc and its chord (see Figure 3.6). The radius and angle of a dome are the same as the corresponding cone.

Monotonic Dome Strategy

Let AB be a chord of angle α on a unit circle. Let D be the dome generated by AB. Without loss of generality, assume AB is a horizontal segment (see Figure 3.6). We use the **Monotonic Rectangle Strategy** to present a wake-up strategy with the above-mentioned makespan. Let the initial robot be located in the right corner of the dome (Point B in Figure 3.6). The initial active robot does a sweep perpendicular to the chord of the dome AB, to wake-up the first inactive robot. Assume the perpendicular line to the chord AB, at the position of the closest robot to B, intersects the curve of the dome at point C. Set the angle $\angle COB = \beta$. Next, these two active robots consider a rectangle of width $W = \text{chord}(\alpha) - \text{chord}(\beta)\cos\frac{\alpha-\beta}{2}$ and height $H = 1 - \cos\alpha/2$ enclosing the remaining region (see Figure 3.6). Then, they follow a strategy as in **Monotonic Rectangle Strategy** (Corollary 3.2) to wake up the enclosing rectangle of the remaining region.

Lemma 3.6. *The **Monotonic Dome Strategy** wakes up inactive robots in a dome of angle α and radius one in a time at most $\alpha/2 + \sin(\alpha/2) + \phi(1 - \cos(\alpha/2))$ with an active robot on*

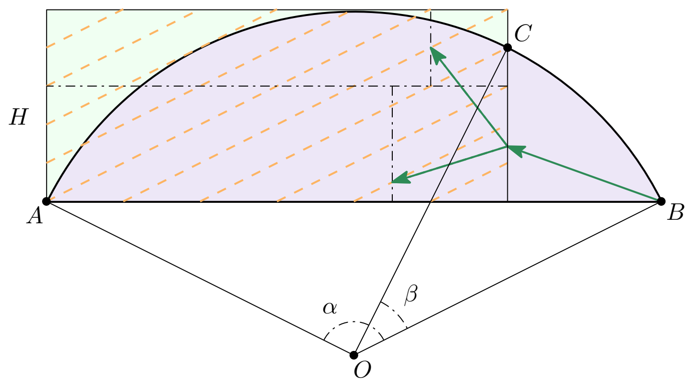


Fig. 3.6: A dome of angle α. Illustration of proof of Lemma 3.6. Solving a dome of angle α using an enclosing rectangle.

the corner of the dome.

Proof. The makespan, dome(α), is bounded by:

$$\begin{aligned}
\mathrm{dome}(\alpha) &\le \|BC\| + W + \phi H \\
&= \mathrm{chord}(\beta) + \mathrm{chord}(\alpha) - \mathrm{chord}(\beta)\cos\frac{\alpha-\beta}{2} + \phi(1-\cos\alpha/2) \\
&= 2\sin\frac{\beta}{2} + 2\sin\alpha/2 - 2\sin\frac{\beta}{2}\cos\frac{\alpha-\beta}{2} + \phi(1-\cos\alpha/2) \\
&= 2\sin\frac{\beta}{2} + \sin(\alpha/2-\beta) + \sin\alpha/2 + \phi(1-\cos\alpha/2) \\
&\le \alpha/2 + \sin\alpha/2 + \phi(1-\cos\alpha/2).
\end{aligned}$$

□

We define *semi-cone* to be the region enclosed between two chords of a unit disk with one common endpoint that does not contain the origin in its interior (see Figure 3.7(a)). We call the common endpoint of two chords the apex of the semi-cone. Let a and b be the length of two chords where $a \le b$ and α be the angle between them. To upper bound the makespan of inactive robots with one active robot on the apex, one can simply enclose a semi-cone with a larger cone and apply the **Monotonic Cone Strategy** (Lemma 3.1) to obtain a makespan of $b + b\alpha\phi$ (see Figure 3.7 (a)). In the following lemma, we prove an upper bound of $b + a\alpha\phi$ when $\alpha < \frac{\pi}{4}$.

Monotonic Semi-cone Strategy

Let Λ be a semi-cone of angle α and chords of length a and b where $a < b$. Let C be the apex of Λ and AC and AB be the chords of length a and b respectively (see Figure 3.7 (b)). Let Λ' be the cone contained in Λ with apex C, angle α and radius CA. Let A' be

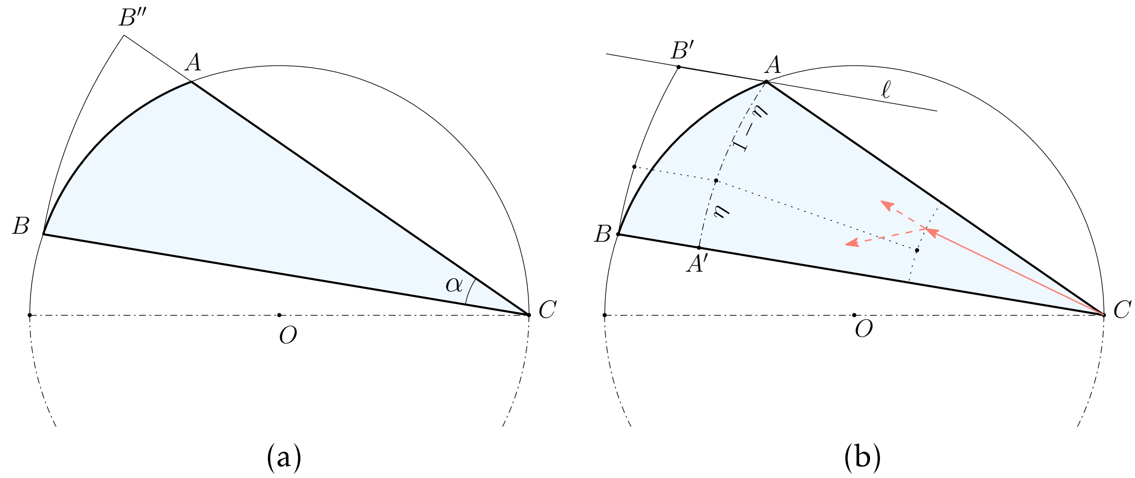


Fig. 3.7: Illustration of **Monotonic Semi-cone Strategy**. (a) a semi-cone. (b) bounding a semi-cone with a simpler geometric object.

the third vertex of Λ' on the segment BC. Consider a line ℓ parallel to CB that goes through A. Let B' be a point on line ℓ outside the underlying unit disk such that the distance between B' and A is the same as the length of the segment BA'. Consider a translation of curve AA' with endpoints B and B'. Since $\alpha < \frac{\pi}{4}$, the curve BB' is outside Λ. Note that geometric object $ACBB'$ encloses Λ.

We use the monotonic wake-up strategy along the segment BC. As before, the first active robot starts by finding the closest inactive robot using a circular sweep. After activating the second robot, they split the remaining unexplored region, R, into two parts with a ratio $\eta : (1-\eta)$ such that the part with coefficient $1-\eta$ contains both of the active robots. Consider the part of R that is contained in Λ'. We split $R \cap \Lambda'$ as in the **Monotonic Cone Strategy** between two robots. The part of the region R that is not contained in Λ' is split by ratio $\eta : (1-\eta)$ using a segment parallel to BC. Next, each of the robots starts to take care of one part. That is one of the robots activates the robots at the parts with coefficient η and the other one activates the robots at the parts with coefficient $(1-\eta)$. They follow a similar approach to find the next closest robot in their region and split the corresponding region according to the same ratio (see Figure 3.7 (b)).

Lemma 3.7. *The **Monotonic Semi-cone Strategy** wakes up robots in a semi-cone of angle $0 < \alpha < \frac{\pi}{4}$ and side lengths a and b where $0 < a < b$, in time at most $b + a\alpha\phi$, where the initial active robot starts at the apex of the semi-cone.*

Proof. Let X_1 and Y_1 be two arbitrary points in Λ'. Using Observation 3.1, we can upper bound the length of segment X_1Y_1 by the sum of a radial distance along the segment BC and an angular distance along the curve AA'. Let X_2 and Y_2 be two points located in the region $ACBB'$ but outside Λ'. Again, we can upper bound the length of segment X_2Y_2 by the sum of a radial distance along the segment BC and an angular

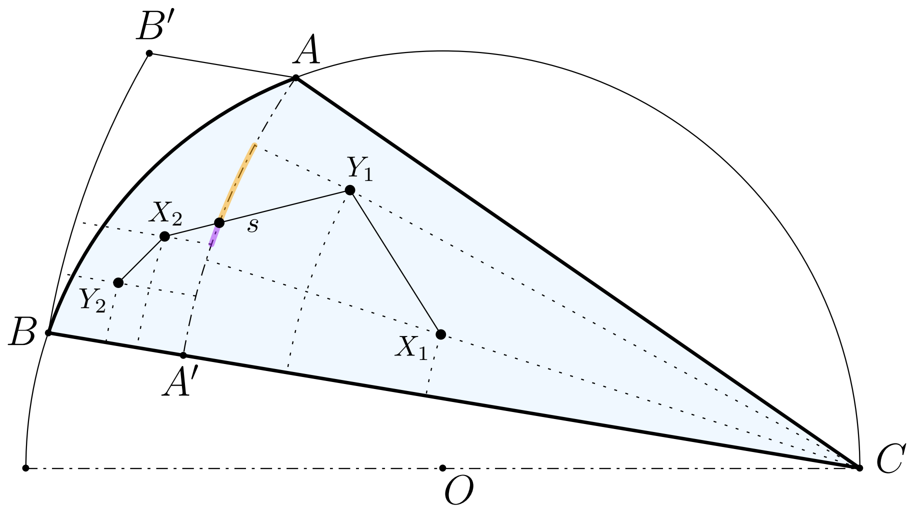


Fig. 3.8: Illustration of proof of Lemma 3.7. Projection of two sub-segments of s on AA' are depicted with orange and purple.

distance along the curve AA'. For this purpose, we project the points, X_2 and Y_2, into segment BA' and curve AA' parallel to curve AA' and segment BC, respectively, to obtain an equivalent angular and radial distance decomposition.

Note that the movement of each robot in the region $ACBB'$ consists of a chain of segments. Moreover, by the orientation of the sweep, it is monotonic along the segment BC of length b. Therefore for each robot, there is at most one segment of movement, say s, that has one endpoint inside Λ' and one endpoint outside Λ'. However, to upper bound the length of s, we split s into two sub-segments at the place where it crosses the curve AA'. Then for each sub-segment, we use the appropriate upper bound from the preceding explanation, depending on which region it is located. Since each sub-segment appears in two different sides of perpendicular line to the intersection of s and the curve AA', the angular projections of two parts of s does not overlap on the curve AA' (see Fig. 3.8).

In summary, the total radial movement of each robot is at most the length of BC, and the total angular movement of each robot, with the same analysis as in the **Monotonic Cone Strategy** and **Monotonic Crown Strategy**, is at most equal to the length of the curve AA' times ϕ, i.e., $a\alpha\phi$. Therefore, the wake-up time of the **Monotonic Semi-cone Strategy** is at most $b + a\alpha\phi$. □

By having an upper bound for the makespan of a dome and semi-cone, we are ready to improve the lower bound on the number of robots needed to ensure makespan of $1 + 2\sqrt{2}$. We show that if there are at least 281 inactive robots in the unit then Conjecture 3.1 is correct.

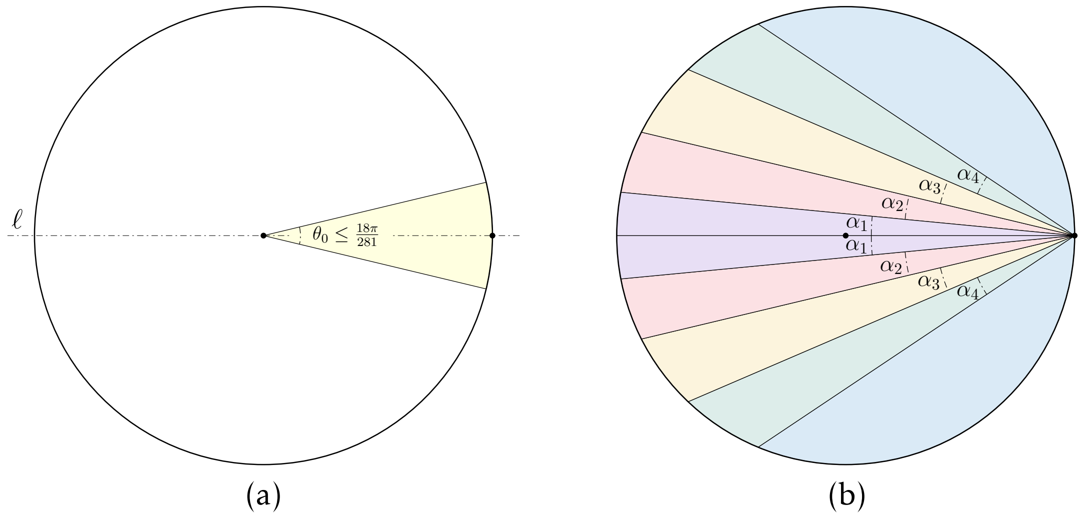


Fig. 3.9: Illustration of proof of Theorem 3.3. (a) stage one. (b) stage two: split into 8 semi-cones and 2 domes.

Large Group Strategy

Note that since $n \geq 281$, there is a cone of the angle at most $\frac{18\pi}{281}$ such that it contains at least 9 inactive robots. As the first stage of the strategy, we recruit a team of at least 10 robots (including the initial active robot). To achieve this goal, the first active robot uses **Monotonic Cone Strategy** to wake up all the active robots in the cone of angle $\frac{18\pi}{281}$. Then, we gather all the active robots in the intersection of the bisector of the cone, namely ℓ, and the boundary of the unit disk (see Figure 3.9(a)).

For the next stage of the strategy, symmetrically, we split each half circle into five regions; four semi-cones and one dome on each side of ℓ (see Figure 3.9(b)). Let the angles of these four semi-cones, as in the Figure 3.9(b), be $\alpha_1 = 0.1252$, $\alpha_2 = 0.1335$, $\alpha_3 = 0.1584$, and $\alpha_4 = 0.2206$. Each robot is assigned to one region and each one needs to wake up their region.

Theorem 3.3. *Let $n \geq 281$ be an integer. The **Large Group Strategy** wakes up n robots in a unit disk in time less than $1 + 2\sqrt{2}$.*

Proof. By Lemma 3.1, the first stage takes time at most $1 + \phi \cdot \frac{18\pi}{281} + \text{chord}(\frac{9\pi}{281}) < 1.4262$. Note that $\text{chord}(\frac{9\pi}{281})$ is the angular contribution of the time to gather the robots at the intersection point.

In the next stage of the strategy, each robot needs to wakes up their region in time at most $2.4022 < 1 + 2\sqrt{2} - 1.4262$ to meet the makespan of $1 + 2\sqrt{2}$. Using **Monotonic Semi-cone Strategy** and Lemma 3.7, the makespan of the first semi-cone of angle α_1 is upper-bounded by $2 + \phi\alpha_1 \cdot \text{chord}(\pi - \alpha_1)$. By setting $\alpha_1 = 0.1252$, we upper bound the makespan of this region with 2.4022. Next, we consider the second semi-cone with angle α_2. Again using **Monotonic Semi-cone Strategy** and Lemma 3.7, the makespan of the second semi-cone is at most $\text{chord}(\pi-\alpha_1)+\phi\alpha_2\cdot\text{chord}(\pi-\alpha_1-\alpha_2)$. By assigning $\alpha_2 = 0.1335$, we also guarantee a makespan less than 2.4022 for the second semi-cone. Similarly, the makespan of at most 2.4022 is ensured for the third semi-cone of angle α_3 and fourth semi-cone of angle α_4 by setting $\alpha_3 = 0.1584$ and $\alpha_4 = 0.2206$. Finally, we are left with a dome of angle $\pi - 2(\alpha_1+\alpha_2+\alpha_3+\alpha_4) < 1.8662$. Using **Monotonic Dome Strategy** and Lemma 3.6, the makespan of a dome of angle 1.8662 is less than 2.4022. □

3.7 A Better Upper-bound On The Wake-Up Time

In this section, we present an improved upper bound for the makespan of $n \in \mathbb{N}$ inactive robots located in a unit disk on Euclidean plane. By a careful study of the first step in the analysis of **Monotonic Crown Strategy** one can propose the following improvement on the analysis of Lemma 3.2 for a unit crown of width w and angle θ.

Improved Monotonic Crown Strategy

Observe that, in the analysis of **Monotonic Crown Strategy** in Lemma 3.2, starting with two robots on the boundary, the angular movement of both robots is upper bounded by the length of the exterior side of the crown. However, the branch of the wake-up tree that is closer to the origin has a smaller angular movement. To take this into account, only for the first step, we consider a different constant than η, say η'. Let two robots be located on the corner of the exterior side of the crown. We follow the same strategy as in the **Monotonic Crown Strategy**. After splitting the crown into two smaller crowns, the part with width ηw is closer to the center. We replace the constant η only in the first step of the analysis of **Monotonic Crown Strategy** with η'.

Lemma 3.8. *The **Improved Monotonic Crown Strategy** wakes up all of the robots in a crown of angle θ and width w starting with two active robots at a corner on the exterior side of the crown in time at most $\theta + \left(\frac{\phi^4}{\phi^3+\theta}\right)w$.*

Proof. For the analysis of the new radial movement, in the branch closer to the origin, we consider a correcting term for the difference of the length of exterior sides, namely $-(1-\eta')w\theta$. Then, we have the following for the total makespan of the crown, $T(\theta, w)$ (compare this formula with the one in the proof of Lemma 3.2):

$$T(\theta,w) \le \theta + \max\{(1+\phi\eta')w - (1-\eta')w\theta,\ (1-\eta')(1+\phi)w\}.$$

Note that $(1+\phi\eta')w - (1-\eta')w\theta$ is increasing and $(1-\eta')(1+\phi)w$ is decreasing with respect to η'. The extreme case happens when the makespan of two branches of the wake-up tree is equal. This is ensured by setting $\eta' = \frac{\phi+\theta}{2\phi+\theta+1}$. Therefore, as an improved upper bound for the makespan of a crown with two active robots located at

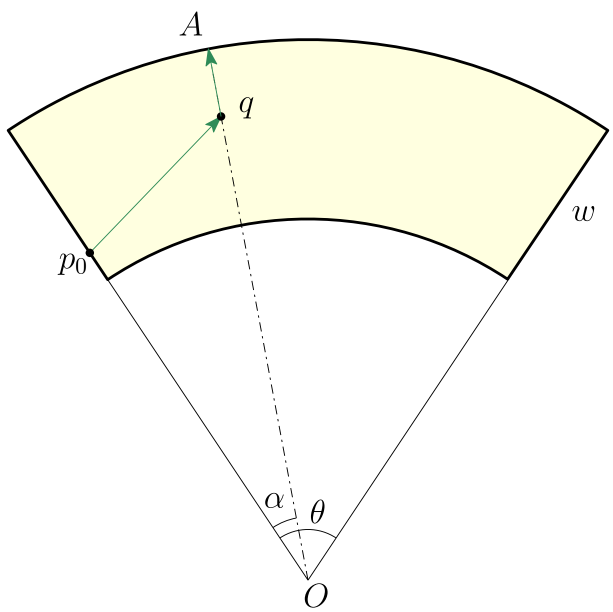


Fig. 3.10: Illustration of Corollary 3.3. The first active robot wakes up the closest inactive robot in an angular sweep and they position themselves at A to apply the **Improved Monotonic Crown Strategy**.

the exterior corner, we have:

$$T(\theta, w) \leq \theta + \left(\frac{\phi^4}{\phi^3 + \theta}\right) w.$$

□

Similar to Corollary 3.1, one can obtain the following improved corollary for the makespan of robots starting with one active robot at a corner of the interior side of the crown.

Improved Monotonic Crown Strategy II

Let O be the center of the unit disk defining the crown. Let q be the first inactive robot in an angular sweep starting from the active robot p_0. Let the angular distance of the active robot p_0 from the robot q be α. Let A be the intersection point of the line passing through O and q with the exterior side of the crown. The robot p_0 travels to the position of the robot q. After activating the robot q, both of them move to point A on the exterior side of the crown to apply the strategy as in **Improved Monotonic Crown Strategy** (see Fig. 3.10).

Corollary 3.3. *The **Improved Monotonic Crown Strategy II** wakes up all of the robots in a crown of angle $\theta \leq \pi$ and width w starting with one active robot at a corner on the interior side of the crown in time at most* $\mathrm{crown}(w, \theta) \leq \theta + \left(1 + \frac{\phi^4}{\phi^3+\theta}\right) w$.

Proof. By the analysis of the length of first two segments that robots take to get in the conditions of **Improved Monotonic Crown Strategy** and the guarantee from Lemma 3.8, we upper bound the total makespan with

$$\begin{aligned}\text{crown}(w,\theta) &\le \|p_0q\| + \|qA\| + \text{crown}(\theta-\alpha, w)\\ &\le \alpha\cdot(1-w) + w + (\theta-\alpha) + \left(\frac{\phi^4}{\phi^3+\theta-\alpha}\right)w\\ &= \theta + \left(1+\frac{\phi^4}{\phi^3+\theta-\alpha} - \alpha\right)w \le \theta + \left(1+\frac{\phi^4}{\phi^3+\theta}\right)w.\end{aligned}$$

To see the last inequality, we define $f(\theta,\alpha) := \left(\frac{\phi^4}{\phi^3+\theta}\right) - \left(\frac{\phi^4}{\phi^3+\theta-\alpha}\right) + \alpha$. Since $\partial f(\theta,\alpha)/\partial\alpha = 1 - \frac{\phi^4}{(\phi^3+\theta-\alpha)^2} > 0$, we have $f(\theta,\alpha) \ge f(\theta,0) = 0$, for $0 \le \alpha \le \theta \le \pi$.

□

In rest of this section, assume that the inactive robots $p_1, p_2, \cdots, p_n$, are in sorted order based on their distance from p_0. That is, if d_i is the distance of p_i from p_0, for each $1 \le i \le n$, then we have $d_1 \le d_2 \le d_3 \le \cdots \le d_n$. We introduce two basic strategies to wake up the robots in set S, and by mixing these two strategies, we get an upper bound for the makespan of n robots located in a unit disk in the plane.

Strategy One

The first active robot at the origin, p_0, travels to the closest robot, p_1, at a distance of d_1, and after activating p_1, both of the robots, p_0 and p_1, travel back to the origin at a total time of $2d_1$. Recall that the point at a distance of d_2 from the origin is p_2. Let ℓ be the line that passes through the origin and p_2. Next, p_0 and p_1 follow different paths. p_0 travels a distance of d_2 to activate p_2. Then, p_0 and p_2 split the remaining region into two equal crowns of angle $\frac{2\pi}{3}$ and width $1-d_2$ (See Figure 3.11). Simultaneously, p_1 uses the **Improved Monotonic Crown Strategy II** to wake up all the robots within the crown of angle $\frac{2\pi}{3}$ and width $1-d_2$ with the bisector ℓ. Using Corollary 3.3, each of p_0, p_1 and p_2 wake up their designated crown in time of at most $\frac{2\pi}{3} + \left(1+\frac{\phi^4}{\phi^3+\frac{2\pi}{3}}\right)(1-d_2)$.

Lemma 3.9. *The* ***Strategy One*** *wakes up inactive robots within a unit disk starting with an active robot in the center in time* $T_1(d_1) \le 1 + 2d_1 + \frac{2\pi}{3} + \left(\frac{\phi^4}{\phi^3+\frac{2\pi}{3}}\right)(1-d_1)$, *where* d_1 *is the distance of closest inactive robot to the initial active robot.*

Proof. Note that $\frac{2\pi}{3} + \left(1+\frac{\phi^4}{\phi^3+\frac{2\pi}{3}}\right)(1-d_2)$ is decreasing as a function of d_2. Since $d_1 \le d_2$, the total makespan of this strategy, $T_1(d_1)$, as a function of d_1, is upper bounded by:

$$T_1(d_1) \le 2d_1 + d_2 + \text{crown}(1-d_2, 2\pi/3)$$

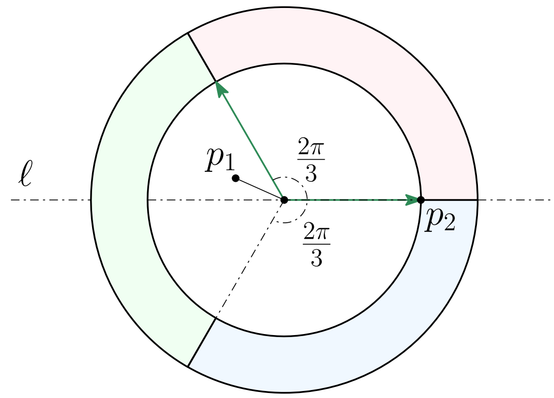


Fig. 3.11: Strategy one. Three crowns of angle $\frac{2\pi}{3}$ and width $1 - d_1$.

$$= 1 + 2d_1 + \frac{2\pi}{3} + \left(\frac{\phi^4}{\phi^3 + \frac{2\pi}{3}}\right)(1 - d_2)$$

$$\leq 1 + 2d_1 + \frac{2\pi}{3} + \left(\frac{\phi^4}{\phi^3 + \frac{2\pi}{3}}\right)(1 - d_1).$$

□

Strategy Two

Note that strategy one is *good* when d_1 is small. We use a simpler idea for the case when d_1 is large, i.e., the robots are close to the boundary of the unit disk. In this strategy, the first active robot, p_0, travels to the closest point p_1 at a distance of d_1. Note that after activating p_1, the disk of radius d_1 centered at the origin has no robots to be activated. Next, p_0 and p_1 split the remaining region into two crowns of angle π and width $1 - d_1$ and they use **Improved Monotonic Crown Strategy II** on these crowns.

Lemma 3.10. *The **Strategy Two** wakes up inactive robots within a unit disk starting with an active robot in the center in time $T_2(d_1) \leq 1 + \pi + \left(\frac{\phi^4}{\phi^3+\pi}\right)(1 - d_1)$, where d_1 is the distance of closest inactive robot to the initial active robot.*

Proof. Using Corollary 3.3 and considering the distance of p_0 from p_1, the total makespan of the second strategy, $T_2(d_1)$, as a function of d_1, is upper bounded by:

$$T_2(d_1) \leq d_1 + \text{crown}(1 - d_1, \pi) \leq 1 + \pi + \left(\frac{\phi^4}{\phi^3 + \pi}\right)(1 - d_1).$$

□

Best of Two Worlds Strategy

By choosing the the best makespan of **Strategy One** and **Strategy Two** depending on the value of d_1, we obtain the following result:

Theorem 3.4. *Let n be a non-negative integer. The* ***Best of Two Worlds Strategy*** *wakes up n robots within a unit disk starting with an active robot in the center in time less than 4.6211.*

Proof. Note that the makespan of the first strategy, $T_1(d_1)$, is upper bounded by an increasing function with respect to d_1 while the upper bound of the makespan for the second strategy, $T_2(d_1)$, is decreasing. By combining the results of the previous two strategies and selecting the best one, namely selecting the smallest of the bounds of $T_1(d_1)$ and $T_2(d_1)$ based on the value of d_1, the bound in the statement of the theorem is achieved. That is $\min_{0 \le d_1 \le 1}\{T_1(d_1), T_2(d_1)\} < 4.6211$. □

Remark 3.1. *Proposition 16 in [16] implies that a wake-up tree of height at most* 4.6211 *can be constructed in linear time.*

Acknowledgement

This research was conducted at LaBRI, University of Bordeaux, during the fall of 2023, while the fourth author was visiting the other authors. The fourth author sincerely appreciates the warm hospitality and the academically enriching atmosphere at LaBRI. Furthermore, gratitude is extended to the University of Ottawa for supporting this visit through the PhD mobility program in France.

Chapter 4

On Rainbow Turán Triangles in Planar Graphs

Prosenjit Bose[1] Vida Dujmović[2] Pat Morin[1] Saeed Odak[2]

Abstract. We study a planar analogue of the multicolour Turán problem. For a sequence of k maximal planar graphs on the same set of n vertices, we aim to determine the smallest value of k such that their union contains a rainbow triangle, i.e., a triangle formed by edges from three distinct graphs. We establish both upper and lower bounds for this problem. We prove that a sequence of at least $0.75n$ planar graphs guarantees the existence of a rainbow triangle. We also provide an almost linear lower bound, using Behrend's construction for three-term arithmetic progression-free sets.

4.1 Introduction

Extremal graph theory is a central area in combinatorics, concerned with understanding the relationship between graph properties and structural constraints. The origins of this field can be traced back to Mantel's theorem, established in 1907, which is one of the first classical results in extremal graph theory. For the graph-theoretic definition, refer to [15] and [26]. All graphs discussed in this chapter are undirected.

Theorem 4.1. *(Mantel [52]) If G is a simple graph on n vertices with $|E(G)| > \frac{n^2}{4}$, then G contains a triangle.*

This theorem addresses the maximum number of edges in a triangle-free graph and sets the foundation for numerous subsequent developments. An example of such developments include Turán's theorem [64], which generalizes Mantel's result to graphs excluding larger complete subgraphs. Turán's theorem determines the maximum number of edges in a graph on n vertices that does not contain a complete subgraph K_r for any $r \geq 3$. It states that the extremal graph achieving this maximum is the $r-1$ partite Turán graph, where the vertex set is partitioned into $r-1$ subsets of sizes as equal

[1]School of Computer Science, Carleton University
[2]School of Electrical Engineering and Computer Science, University of Ottawa

as possible. The number of edges in this graph is given by $\left(1-\frac{1}{r-1}+o(1)\right)\frac{n^2}{2}$. When $r=3$, this reduces to Mantel's theorem, which describes the triangle-free case. These results form the core of classical extremal problems, where the goal is to determine the maximum number of edges in graphs avoiding certain substructures.

Over the years, researchers have extended Mantel's theorem to various settings, including colored graphs and multigraphs [44, 49]. One notable extension involves studying *rainbow triangles*, where edges are drawn from distinct graphs in a sequence. Let $k \geq 3$ be a positive integer. Let $G_1, G_2, \cdots, G_k$ be a sequence of graphs on the same set V of vertices. We define a *rainbow triangle* in the sequence $G_1, G_2, \ldots, G_k$ to be three distinct vertices $u, v, w \in V$, so that $\{u,v\} \in E(G_a)$, $\{v,w\} \in E(G_b)$, and $\{w,u\} \in E(G_c)$ for distinct indices $a, b, c \in \{1, 2, \cdots, k\}$. A rainbow version of Mantel's Theorem is studied by Aharoni et al. [1].

Theorem 4.2. *(Aharoni et al. [1]) Let G_1, G_2, G_3 be graphs on a common set of n vertices. If $|E(G_i)| > \frac{26-2\sqrt{7}}{81}n^2 (\approx 0.257n^2)$ for $1 \leq i \leq 3$, then there exists a rainbow triangle in the sequence G_1, G_2, G_3. Moreover, the constant $\frac{26-2\sqrt{7}}{81}$ is the best possible.*

Recall that planar graphs are graphs that can be embedded in the plane without any edge crossings. In such an embedding, a face is a maximal connected region bounded by edges, including the unbounded outer region. A triangulation is a planar graph in which every face (including the outer one) is bounded by exactly three edges—that is, an edge-maximal planar graph.

Both Mantel's theorem and its rainbow counterpart establish a connection between edge density and the existence of (rainbow) triangles. These results motivate further exploration in various constrained settings, such as planar graphs. Since edge-maximal planar graphs on n vertices have at most $3n-6$ edges, planarity imposes inherent restrictions on edge density. This limitation introduces a new dimension to the problem rather than focusing on edge counts, we examine how many planar graphs are needed to guarantee the emergence of a rainbow triangle.

The interplay between combinatorial density and geometric constraints in planar graphs raises natural and intriguing extremal questions that have not been fully explored. Our work addresses this gap by investigating the threshold number of maximal planar graphs required to force a rainbow triangle in their union.

Problem 4.1. *Let $G_1, G_2, \ldots, G_k$ be a sequence of (not necessarily distinct) maximal planar graphs on the same set of n vertices. What is the smallest value of k such that the sequence $G_1, G_2, \ldots, G_k$ is guaranteed to contain a rainbow triangle?*

We present a simple linear upper bound (see Theorem 4.3) and an almost linear lower bound for the value of k as a function of n (see Theorem 4.5).

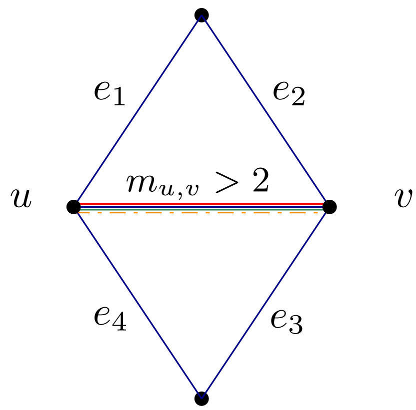


Fig. 4.1: Illustration of proof of Theorem 4.3.

4.2 Upper Bound

In this section, as an upper bound, we show that any sequence of at least $0.75n$ maximal planar graphs on the same vertex set of size n contains a rainbow triangle. Formally, we state the following theorem.

Theorem 4.3. *Let $n > 3$ be a positive integer. Let $G_1, G_2, G_3, \cdots, G_k$ be a sequence of maximal planar graphs on the same set of n vertices. If $k \geq \frac{3n}{4}$, then the sequence $G_1, G_2, \ldots, G_k$ contains a rainbow triangle.*

Proof. Let $G_1, G_2, \cdots, G_k$ be a sequence of maximal planar graphs for some $k \in \mathbb{N}$ such that $k \geq \frac{3n}{4}$. Assume there is no rainbow triangle in the sequence $G_1, G_2, \ldots, G_k$. For each two distinct vertices $u, v \in V$, let $m_{u,v}$ be the number of graphs in the sequence $G_1, G_2, \cdots, G_k$ that contain $\{u, v\}$ as an edge. That is if $A_{u,v} = \{G_i | \ 1 \leq i \leq k \text{ and } \{u, v\} \in E(G_i)\}$, then $m_{u,v} = |A_{u,v}|$. Let $E_{\leq 2} := \{\{u, v\} \in \binom{V}{2} : m_{u,v} \leq 2\}$ and $E_{>2} := \binom{V}{2} \setminus E_{\leq 2}$.[3]

Let $\{u, v\}$ be a pair such that $m_{u,v} > 2$. Fix a graph $H \in A_{u,v}$. Since H is a triangulation, there is a unique embedding of H is the plane. Therefore, there are two faces of H incident to the edge $\{u, v\}$. Let e_1, e_2, e_3, and e_4 be the four edges of H incident to these two faces, other than the edge $\{u, v\}$. Since there is no rainbow triangle in the sequence $G_1, G_2, \ldots, G_k$, none of the edges e_1, e_2, e_3, e_4 appear in any of the graphs $\{G_1, G_2, \cdots, G_k\} \setminus \{H\}$. Therefore, the edges e_1, e_2, e_3, e_4 only appear in $E(H)$ (refer to Fig. 4.1).

Note that since each G_i $(1 \leq i \leq k)$ is a triangulation, for each e_j $(1 \leq j \leq 4)$ there can be at most one more pair $\{z, w\} \neq \{u, v\}$ such that $m_{z,w} > 2$, $H \in A_{z,w}$, and e_j is the incident to the edge $\{z, w\}$. Therefore, we have $4 \sum_{e \in E_{>2}} m_e \leq 2\binom{n}{2}$.

The total number of edges in the sequence $G_1, G_2, \ldots, G_k$ is $k \cdot (3n - 6) = \sum_{\{u,v\} \in \binom{V}{2}} m_{u,v}$. On the other hand, $\sum_{e \in E_{\leq 2}} m_e \leq 2 \cdot \binom{n}{2}$. Therefore,

[3]Note that $\binom{V}{2} = \{\{u, v\} : u, v \in V \text{ and } u \neq v\}$.

$$k \cdot (3n-6) = \sum_{\{u,v\} \in \binom{V}{2}} m_{u,v} \leq \frac{5}{2}\binom{n}{2} = \frac{5}{4}n(n-1).$$

$$k \leq \frac{5(n^2-n)}{4(3n-6)} < \frac{3n}{4} \quad .$$

This contradicts the fact that $k \geq \frac{3n}{4}$. □

4.3 Lower Bound

In this section, we construct a near-linear number of maximal planar graphs on the same set of vertices without creating a rainbow triangle. The main tool in our lower bound argument is the so-called Behrend's construction on three-term arithmetic progress-free sequence [8]. A set A *avoids* the t variable equation $g(x_1, x_2, \ldots, x_t) = 0$ if there are no pairwise distinct t elements $a_1, a_2, \ldots, a_t \in A$ such that $g(a_1, a_2, \ldots, a_t) = 0$. We say that a set A of integers is a *three-term arithmetic progression-free* if there are no three distinct elements $x, y, z \in A$ such that $x + z = 2y$ (the set A avoids $x + z - 2y = 0$). Behrend's construction gives a lower bound on the size of a largest three-term arithmetic progression-free subset of $\{1, 2, 3, \cdots, N\}$ for every positive integer N.

Theorem 4.4. *(Behrend [8]) There exists a constant $C > 0$ such that for every positive integer N, there exists a three-term arithmetic progression-free set $A \subseteq \{1, 2, \ldots, N\}$ with $|A| \geq N 2^{-C\sqrt{\log N}}$.*

The following theorem presents the main result of this section.

Theorem 4.5. *There exist $k = \Omega(n/2^{O(\sqrt{\log n})})$ maximal planar graphs $G_1, G_2, \cdots, G_k$ on the same vertex set of size n such that the sequence $G_1, G_2, \ldots, G_k$ contains no rainbow triangle.*

Proof. Let x be a positive integer. The maximal planar graph $G_x^\star$ is defined on the vertex set $V_x^\star = \{0, 1, 2, \cdots, 3x-1\}$ with the edge set

$$E_x^\star := \{\{3i+j, 3i+j+3\} \colon 0 \leq i < x-1 \text{ and } 0 \leq j \leq 2\} \cup$$
$$\{\{3i, 3i+1\} \colon 0 \leq i < x\} \cup \{\{3i+1, 3i+2\} \colon 0 \leq i < x\} \cup \{\{3i+2, 3i\} \colon 0 \leq i < x\} \cup$$
$$\{\{3i, 3i+4\} \colon 0 \leq i < x-1\} \cup \{\{3i+1, 3i+5\} \colon 0 \leq i < x-1\} \cup \{\{3i+2, 3i+3\} \colon 0 \leq i < x-1\}.$$

$G_3^\star$ is depicted in Fig. 4.2(a) and (b). Let p be a large prime number. As a lower bound construction, we introduce a sequence of graphs that contains only graphs isomorphic to $G_p^\star$. Let $n = 3p$ and let $V = \{0, 1, 2, \cdots, n-1\}$. Let $S = \{i : 1 \leq i \leq p/4$ and $i \neq 0 \pmod 3\}$. For each $s \in S$, we define the graph G_s on V isomorphic to $G_p^\star$ using the function $\psi : V_p^\star \to V$ with the following definition as an isomorphism between $G_p^\star$ and G_s.

a,b,c	a,b,c	a,b,c	a,b,c
1,1,1	1,3,3	2,3,1	3,3,3
1,1,2	1,3,4	2,3,2	3,3,4
1,1,3	1,4,1	2,3,3	3,4,1
1,1,4	1,4,2	2,3,4	3,4,2
1,2,1	1,4,3	2,4,1	3,4,3
1,2,2	1,4,4	2,4,2	3,4,4
1,2,3	2,2,1	2,4,3	4,4,1
1,2,4	2,2,2	2,4,4	4,4,2
1,3,1	2,2,3	3,3,1	4,4,3
1,3,2	2,2,4	3,3,2	4,4,4

Table 4.1: The list of possible values for (a, b, c)

$$\psi(u) = \begin{cases} 0, & \text{if } u = 0, \\ \psi(u-1) + s \pmod{3p} & \text{otherwise.} \end{cases}$$

Since p is a prime number and no element of S is a multiple of 3, observe that the defined embedding of G_s in $V_p^\star$ has the following property. For each edge $\{u,v\} \in E(G_s)$, we have $|v-u| \in \{s, 2s, 3s, 4s\} \pmod{3p}$. This is illustrated in Fig. 4.2(c).

Let Q be the largest subset of S such that the set $\{G_s : s \in Q\}$ does not contain a rainbow triangle. Since elements of S are less than $p/4$, equivalently $Q \subseteq S$ is the largest set, such that the equation $ax + by = cz$ has no solution where x, y, and z are distinct elements of Q and $a,b,c \in \{1,2,3,4\}$. Note that up to symmetry there are 40 possibilities for the values of a, b and c (listed in Table 4.1). To obtain a lower bound on the size of the largest such a set, we use Behrend's construction [8]. Recall that $n = 3p$. By a slight modification of Behrend's construction, we can generate a subset $\mathcal{B} \subseteq \{1,2,\ldots,n\}$ where equations $x+y = 2x$, $x+2y = 3z$, and $x+3y = 4z$ are avoided (Gray colours in Table 4.1) and $|\mathcal{B}| \geq n2^{-C\sqrt{\log n}}$ for some constant $C > 0$.

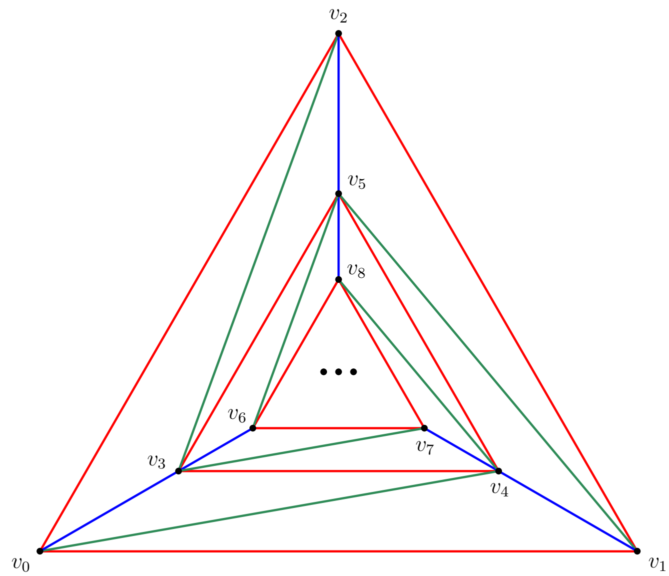
Note that we aim to create a subset $\mathbb{Q} \subset \mathcal{B}$ such that, at the same time, none of the elements of $\mathbb{Q}$ is divisible by 3, and all of the equations in Table 4.1 are avoided. To achieve this, we partition $\mathcal{B}$ using the following function

$$f : \mathbb{B} \to \mathbb{N} \times \mathbb{N} \times \{1,2\} \times \mathbb{N} \times \{1,2,3,4\}$$

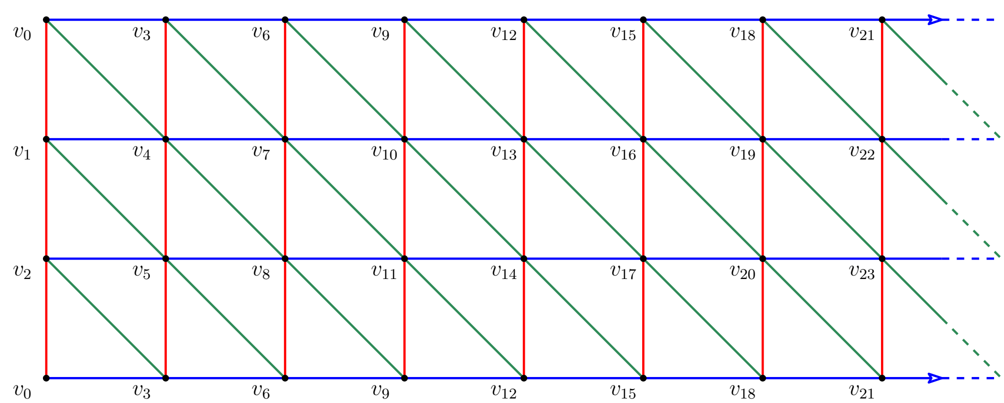
$$f(n) = (\nu_2(n), \nu_3(n), \mathrm{r}_3(n), \nu_5(n), \mathrm{r}_5(n)),$$

where for integer $k > 1$, $\nu_k(n)$ is the largest power of k that divides n and $\mathrm{r}_k(n)$ is $n/k^{\nu_k(n)} \bmod k$.

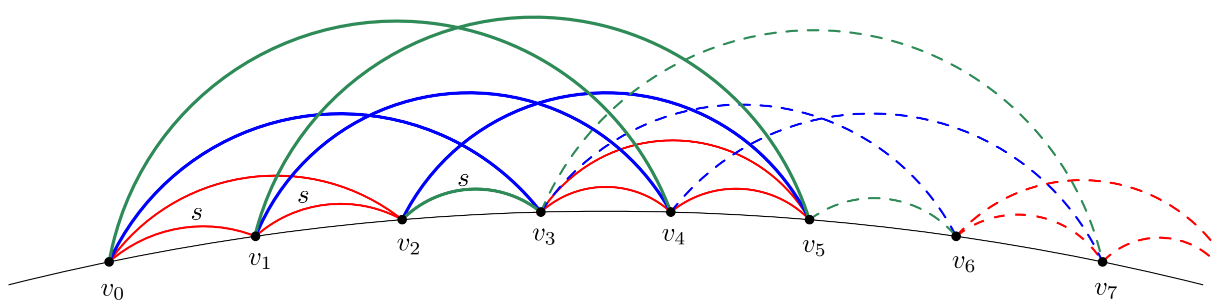
For each tuple t in the range of f, we can generate a set from the pre-image of t, $f^{-1}(t)$, such that it avoids all the equations listed in Table 4.1. Let $X \subseteq \{1,2,\ldots,n\}$



(a)



(b)



(c)

Fig. 4.2: (a) The graph $G_3^\star$. (b) A different representation of $G_i^\star$; the first row of vertices is identified with the last row. (c) The embedding of G_s.

be a set such that $f(x) = t$ for each $x \in X$. Then, by dividing each element of X by $2^{t(1)}3^{t(2)}5^{t(4)}$, we can obtain such a set. Note that in Table 4.1, for equations with the purple colour, we have $\gcd(a,b,c) > 1$, and powers of two in the definition of f (the first element in the tuple) take care of avoiding the equations with the green and blue colours. Powers of three (the second and third elements in the tuple) and powers of five (the last two elements in the tuple) take care of avoiding the equations with the orange and black colours, respectively.

Observe that since $\mathcal{B} \subset \{1,2,\ldots,n\}$, we have $|f(\mathcal{B})| = O(\log^3 n)$. Taking an element from $f(\mathcal{B})$ with the largest pre-image proves the theorem. □

4.4 Conclusion

We demonstrated that a specific maximal planar graph with a vertex set of size n can be embedded in at least $n^{1-o(1)}$ distinct ways on a labeled set of size n without forming a rainbow triangle. Conversely, we proved that any sequence of maximal planar graphs of length at least $0.75n$ must contain a rainbow triangle.

In addition to the graph-theoretic results, our work also connects to a classic problem in number theory. To build our lower bound, we use special sets of numbers that avoid three-term arithmetic progressions—sequences like $a, a+d, a+2d$. These sets, first studied by Behrend [8], help us choose which edges to include in each planar graph so that no rainbow triangle is formed. This connection shows how ideas from number theory can be useful in problems about planar graphs, and it opens up new directions for combining techniques from different areas of mathematics. Due to the construction of the lower bound example in Section 4.3 and its connection to the three term arithmetic progression problem, the authors believe that the upper bound should be $o(n)$. Closing the gap between these upper and lower bounds remains an open problem for future research.

Bibliography

[1] Ron Aharoni, Matt DeVos, Sebastián González Hermosillo de la Maza, Amanda Montejano, and Robert Šámal. A rainbow version of Mantel's theorem. *Advances in Combinatorics*, 2020.

[2] Stephen Alstrup, Cyril Gavoille, Haim Kaplan, and Theis Rauhe. Nearest common ancestors: A survey and a new algorithm for a distributed environment. *Theory Comput. Syst.*, 37(3):441–456, 2004.

[3] Esther M. Arkin, Michael A. Bender, Sándor P Fekete, Joseph S. B. Mitchell, and Martin Skutella. The freeze-tag problem: how to wake up a swarm of robots. *Algorithmica*, 46:193–221, 2006.

[4] Esther M. Arkin, Michael A. Bender, and Dongdong Ge. Improved approximation algorithms for the freeze-tag problem. In Arnold L. Rosenberg and Friedhelm Meyer auf der Heide, editors, *SPAA 2003: Proceedings of the Fifteenth Annual ACM Symposium on Parallelism in Algorithms and Architectures, June 7-9, 2003, San Diego, California, USA (part of FCRC 2003)*, pages 295–303. ACM, 2003.

[5] Francisco Arrepol, Patricio Asenjo, Raúl Astete, Víctor Cartes, Anahí Gajardo, Valeria Henríquez, Catalina Opazo, Nicolás Sanhueza-Matamala, and Christopher Thraves Caro. Separating path systems in trees. *arXiv preprint arXiv:2306.00843*, 2023.

[6] Patrick Assouad. Densité et dimension. *Annales de l'Institut Fourier*, 33(3):233–282, 1983.

[7] József Balogh, Béla Csaba, Ryan R. Martin, and András Pluhár. On the path separation number of graphs. *Discret. Appl. Math.*, 213:26–33, 2016.

[8] Felix A Behrend. On sets of integers which contain no three terms in arithmetical progression. *Proceedings of the National Academy of Sciences*, 32(12):331–332, 1946.

[9] Michael A. Bekos, Giordano Da Lozzo, Petr Hliněný, and Michael Kaufmann. Graph product structure for h-framed graphs. *CoRR*, abs/2204.11495, 2019.

[10] Michael A. Bender and Martin Farach-Colton. The LCA problem revisited. In Gaston H. Gonnet, Daniel Panario, and Alfredo Viola, editors, *LATIN 2000: Theoretical Informatics, 4th Latin American Symposium, Punta del Este, Uruguay, April 10-14, 2000, Proceedings*, volume 1776 of *Lecture Notes in Computer Science*, pages 88–94. Springer, 2000.

[11] Omer Berkman and Uzi Vishkin. Recursive star-tree parallel data structure. *SIAM J. Comput.*, 22(2):221–242, 1993.

[12] Ahmad Biniaz, Prosenjit Bose, Jean-Lou De Carufel, Anil Maheshwari, Babak Miraftab, Saeed Odak, Michiel Smid, Shakhar Smorodinsky, and Yelena Yuditsky. On separating path and tree systems in graphs. *Discret. Math. Theor. Comput. Sci.*, 27(2), 2025.

[13] Béla Bollobás and Alex D. Scott. On separating systems. *Eur. J. Comb.*, 28(4):1068–1071, 2007.

[14] Marthe Bonamy, Fábio Botler, François Dross, Tássio Naia, and Jozef Skokan. Separating the edges of a graph by a linear number of paths. *Advances in Combinatorics*, 6:7, 2023.

[15] John Adrian Bondy, Uppaluri Siva Ramachandra Murty, et al. *Graph theory with applications*, volume 290. Macmillan London, 1976.

[16] Nicolas Bonichon, Arnaud Casteigts, Cyril Gavoille, and Nicolas Hanusse. Freeze-tag in L_1 has wake-up time five with linear complexity. In Dan Alistarh, editor, *38th International Symposium on Distributed Computing, DISC 2024, October 28 to November 1, 2024, Madrid, Spain*, volume 319 of *LIPIcs*, pages 9:1–9:16. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2024.

[17] Nicolas Bonichon, Cyril Gavoille, Nicolas Hanusse, and Saeed Odak. Euclidean freeze-tag problem on plane. *36th Canadian Conference on Computational Geometry (CCCG)*, 2024.

[18] Prosenjit Bose, Vida Dujmović, Mehrnoosh Javarsineh, and Pat Morin. Asymptotically optimal vertex ranking of planar graphs. *CoRR*, abs/2007.06455, 2020.

[19] Prosenjit Bose, Pat Morin, and Saeed Odak. An optimal algorithm for product structure in planar graphs. In Artur Czumaj and Qin Xin, editors, *18th Scandinavian Symposium and Workshops on Algorithm Theory, SWAT 2022, June 27-29, 2022, Tórshavn, Faroe Islands*, volume 227 of *LIPIcs*, pages 19:1–19:14. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2022.

[20] André Bouchet. Orientable and nonorientable genus of the complete bipartite graph. *J. Comb. Theory, Ser. B*, 24(1):24–33, 1978.

[21] Josh Brunner and Julian Wellman. An optimal algorithm for online freeze-tag. In Martin Farach-Colton, Giuseppe Prencipe, and Ryuhei Uehara, editors, *10th International Conference on Fun with Algorithms, FUN 2021, May 30 to June 1, 2021, Favignana Island, Sicily, Italy*, volume 157 of *LIPIcs*, pages 8:1–8:11. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2021.

[22] Mao-cheng Cai. On separating systems of graphs. *Discret. Math.*, 49(1):15–20, 1984.

[23] Bernard Chazelle and Emo Welzl. Quasi-optimal range searching in spaces of finite vc-dimension. *Discrete & Computational Geometry*, 4:467–489, 1989.

[24] Lucas de Oliveira Silva. Freeze-tag is NP-hard in 3D with L_1 distance. *CoRR*, abs/2301.07757, 2023.

[25] Michal Debski, Stefan Felsner, Piotr Micek, and Felix Schröder. Improved bounds for centered colorings. In Shuchi Chawla, editor, *Proceedings of the 2020 ACM-SIAM Symposium on Discrete Algorithms, SODA 2020, Salt Lake City, UT, USA, January 5-8, 2020*, pages 2212–2226. SIAM, 2020.

[26] Reinhard Diestel. *Graph Theory, Fifth Edition*, volume 173 of *Graduate texts in mathematics*. Springer, 2017.

[27] Vida Dujmović, Gwenaël Joret, Piotr Micek, Pat Morin, Torsten Ueckerdt, and David R. Wood. Planar graphs have bounded queue-number. *J. ACM*, 67(4):22:1–22:38, 2020.

[28] Vida Dujmovic, Pat Morin, and David R. Wood. Graph product structure for non-minor-closed classes. *CoRR*, abs/1907.05168, 2019.

[29] Vida Dujmović, Louis Esperet, Cyril Gavoille, Gwenaël Joret, Piotr Micek, and Pat Morin. Adjacency labelling for planar graphs (and beyond). *J. ACM*, 68(6):42:1–42:33, 2021.

[30] Vida Dujmović, Louis Esperet, Gwenaël Joret, Bartosz Walczak, and David R. Wood. Planar graphs have bounded nonrepetitive chromatic number. *CoRR*, abs/1904.05269, 2019.

[31] M. Rudoy E. D. Demaine. Freeze tag is hard in 3D. *In Abstracts from the 27th Fall Workshop on Computational Geometry*, page 108–110, 2017.

[32] Louis Esperet, Gwenaël Joret, and Pat Morin. Sparse universal graphs for planarity. *CoRR*, abs/2010.05779, 2020.

[33] Victor Falgas-Ravry, Teeradej Kittipassorn, Dániel Korándi, Shoham Letzter, and Bhargav P. Narayanan. Separating path systems. *J. Comb.*, 5(3):335–354, 2014.

[34] Johannes Fischer and Volker Heun. Theoretical and practical improvements on the rmq-problem, with applications to LCA and LCE. In Moshe Lewenstein and Gabriel Valiente, editors, *Combinatorial Pattern Matching, 17th Annual Symposium, CPM 2006, Barcelona, Spain, July 5-7, 2006, Proceedings*, volume 4009 of *Lecture Notes in Computer Science*, pages 36–48. Springer, 2006.

[35] Florent Foucaud, Eleonora Guerrini, Matjaz Kovse, Reza Naserasr, Aline Parreau, and Petru Valicov. Extremal graphs for the identifying code problem. *Eur. J. Comb.*, 32(4):628–638, 2011.

[36] Florent Foucaud and Matjaž Kovše. Identifying path covers in graphs. *J. Discrete Algorithms*, 23:21–34, 2013.

[37] Alan M. Frieze, Ryan R. Martin, Julien Moncel, Miklós Ruszinkó, and Cliff Smyth. Codes identifying sets of vertices in random networks. *Discret. Math.*, 307(9-10):1094–1107, 2007.

[38] Fănică Gavril. The intersection graphs of subtrees in trees are exactly the chordal graphs. *Journal of Combinatorial Theory, Series B*, 16:47–56, 1974.

[39] Dániel Gerbner and Géza Tóth. Separating families of convex sets. *Comput. Geom.*, 46(9):1056–1058, 2013.

[40] Valentin Gledel and Aline Parreau. Identification of points using disks. *Discret. Math.*, 342(1):256–269, 2019.

[41] Mikael Hammar, Bengt J. Nilsson, and Mia Persson. The online freeze-tag problem. In José R. Correa, Alejandro Hevia, and Marcos A. Kiwi, editors, *LATIN 2006: Theoretical Informatics, 7th Latin American Symposium, Valdivia, Chile, March 20-24, 2006, Proceedings*, volume 3887 of *Lecture Notes in Computer Science*, pages 569–579. Springer, 2006.

[42] Dov Harel and Robert Endre Tarjan. Fast algorithms for finding nearest common ancestors. *SIAM J. Comput.*, 13(2):338–355, 1984.

[43] David Haussler. Sphere packing numbers for subsets of the boolean n-cube with bounded Vapnik-Chervonenkis dimension. *Journal of Combinatorial Theory, Series A*, 69(2):217–232, 1995.

[44] Zhen He, Peter Frankl, Ervin Győri, Zequn Lv, Nika Salia, Casey Tompkins, Kitti Varga, and Xiutao Zhu. Extremal results for graphs avoiding a rainbow subgraph. *Electron. J. Comb.*, 31(1), 2024.

[45] Iiro S. Honkala, Mark G. Karpovsky, and Simon Litsyn. Cycles identifying vertices and edges in binary hypercubes and 2-dimensional tori. *Discret. Appl. Math.*, 129(2-3):409–419, 2003.

[46] Johan Ludwig William Valdemar Jensen. Sur les fonctions convexes et les inégalités entre les valeurs moyennes. *Acta mathematica*, 30(1):175–193, 1906.

[47] Mark G. Karpovsky, Krishnendu Chakrabarty, and Lev B. Levitin. On a new class of codes for identifying vertices in graphs. *IEEE Trans. Inf. Theory*, 44(2):599–611, 1998.

[48] Gyula Katona. On separating systems of a finite set. *Journal of Combinatorial Theory*, 1(2):174–194, 1966.

[49] Peter Keevash, Dhruv Mubayi, Benny Sudakov, and Jacques Verstraëte. Rainbow turán problems. *Combinatorics, Probability and Computing*, 16(1):109–126, 2007.

[50] Jochen Könemann, Asaf Levin, and Amitabh Sinha. Approximating the degree-bounded minimum diameter spanning tree problem. *Algorithmica*, 41(2):117–129, 2005.

[51] André Kündgen, Dhruv Mubayi, and Prasad Tetali. Minimal completely separating systems of k-sets. *J. Comb. Theory, Ser. A*, 93(1):192–198, 2001.

[52] W Mantel. Solution to problem 28, by h. *Gouwentak, W. Mantel, J. Teixeira de Mattes, F. Schuh, and WA Wythoff, Wiskundige Opgaven*, 10:60–61, 1907.

[53] Jiri Matousek. *Lectures on discrete geometry*, volume 212. Springer Science & Business Media, 2013.

[54] Pat Morin. A fast algorithm for the product structure of planar graphs. *Algorithmica*, 83(5):1544–1558, 2021.

[55] Lehilton Lelis Chaves Pedrosa and Lucas de Oliveira Silva. Freeze-tag is np-hard in 3d with l_1 distance. In Cristina G. Fernandes and Sergio Rajsbaum, editors, *Proceedings of the XII Latin-American Algorithms, Graphs and Optimization Symposium, LAGOS 2023, Huatulco, Mexico, September 18-22, 2023*, volume 223 of *Procedia Computer Science*, pages 360–366. Elsevier, 2023.

[56] A Rényi. On random generating elements of a finite boolean algebra. *Acta Sci. Math. Szeged*, 22(75-81):4, 1961.

[57] Gerhard Ringel. Das geschlecht des vollständigen paaren graphen. In *Abhandlungen aus dem Mathematischen Seminar der Universität Hamburg*, volume 28(3-4), pages 139–150. Springer, 1965.

[58] Gerhard Ringel. Der vollständige paare graph auf nichtorientierbaren flächen. *Journal für die reine und angewandte Mathematik*, 220:88–93, 1965.

[59] Petri Rosendahl. On the identification of vertices using cycles. *Electron. J. Comb.*, 10, 2003.

[60] Norbert Sauer. On the density of families of sets. *Journal of Combinatorial Theory, Series A*, 13(1):145 – 147, 1972.

[61] Baruch Schieber and Uzi Vishkin. On finding lowest common ancestors: Simplification and parallelization. *SIAM J. Comput.*, 17(6):1253–1262, 1988.

[62] Saharon Shelah. A combinatorial problem; stability and order for models and theories in infinitary languages. *Pacific J. Math.*, 41(1):247–261, 1972.

[63] Marcelo O. Sztainberg, Esther M. Arkin, Michael A. Bender, and Joseph S. B. Mitchell. Theoretical and experimental analysis of heuristics for the "freeze-tag" robot awakening problem. *IEEE Trans. Robotics*, 20(4):691–701, 2004.

[64] Paul Turán. On an external problem in graph theory. *Mat. Fiz. Lapok*, 48:436–452, 1941.

[65] Torsten Ueckerdt, David R. Wood, and Wendy Yi. An improved planar graph product structure theorem. *CoRR*, abs/2108.00198, 2021.

[66] Vladimir N. Vapnik and Alexei Ya. Chervonenkis. On the uniform convergence of relative frequencies of events to their probabilities. *Theory of Probability and its Applications*, 16(2):264–280, 1971.

[67] Ingo Wegener. On separating systems whose elements are sets of at most k elements. *Discret. Math.*, 28(2):219–222, 1979.

[68] Belinda Wickes. Separating path systems for the complete graph. *arXiv preprint arXiv:2209.04302*, 2022.

[69] Gábor Wiener, Éva Hosszu, and János Tapolcai. On separating systems with bounded set size. *Discret. Appl. Math.*, 276:172–176, 2020.

[70] Ehsan Najafi Yazdi, Alireza Bagheri, Zahra Moezkarimi, and Hamidreza Keshavarz. An o(1)-approximation algorithm for the 2-dimensional geometric freeze-tag problem. *Inf. Process. Lett.*, 115(6-8):618–622, 2015.

[71] J. Yu Z. Abel, H. A. Akitaya. Freeze tag awakening in 2D is NP-hard. *In Abstracts from the 27th Fall Workshop on Computational Geometry*, pages 105–107, 2017.