Graph Burning

on necklace graphs and cycle-forests

by

A.B. Kooijmans

to obtain the degree of Bachelor of Science
at the Delft University of Technology,
to be defended publicly on Wednesday July 2, 2025 at 13:00 PM.

Student number:	5783623	
Project duration:	April 22, 2025 – July 2, 2025	
Thesis committee:	Dr. Y. Murakami,	TU Delft, supervisor
	Dr. N. D. Verhulst,	TU Delft, supervisor
	Dr. ir. E. G. Rens,	TU Delft

An electronic version of this thesis is available at `http://repository.tudelft.nl/`.

TU Delft

Contents

1 Introduction **2**

2 Preliminaries **4**
2.1 Concepts and definitions from graph theory. 4
2.2 Graph burning 5

3 Literature review **7**
3.1 Burning number. 7
3.2 Algorithms and complexity results 8

4 Necklace graphs **10**

5 Cycle-forests **15**

6 Conclusion & Discussion **17**

Summary

Graph burning is a process that models the spread of information through a network. This process is divided into rounds and is visualized as a spreading fire. Each round, a source of fire may be chosen from where the fire spreads and burns surrounding points. From every burned point, the fire propagates through adjacent points, eventually covering the entire network. The burning number of a graph, $b(G)$, was introduced to measure the time required to burn the entire network.

It was proven that path graphs on n vertices have burning number $\lceil \sqrt{n} \rceil$. It was then conjectured that all connected graphs have at most this burning number. The burning number has been estimated or identified for several classes of graphs. We provide a lower and upper bound for the burning number of a new class of graphs with a path-like structure, namely necklace graphs. Necklace graphs are constructed by concatenating smaller graphs, called pearls. To obtain bounds, we define lower-bound paths and upper-bound paths, which are paths connecting two designated vertices. We show that a lower-bound path always exists in the form of a shortest path, while finding an upper-bound path for an arbitrary necklace is NP-complete. For necklace graphs where a shortest path is an upper-bound path, we show that the burning number can take only two values.

Finally, we consider cycle-forests, which are disconnected graphs whose components are cycle graphs. We prove that burning cycle-forests is NP-complete.

Layman's summary

To understand how quickly information spreads across a network, the graph burning model can be used. Graph burning is a step-by-step process that uses mathematical objects called graphs to model the network. In each step, the information spreads from a point in the network, a vertex, to its neighbors, similar to a fire that spreads to the surrounding environment. The type of network which takes the longest to burn is a path, where all vertices are linked in a sequence.

We use the concept of paths to construct necklace graphs, which have a path-like structure by stringing together so-called pearls. We define two types of paths in a necklace graph, a lower-bound path and an upper-bound path, to estimate how quickly information spreads in a necklace graph. We show that finding a lower-bound path in a necklace is easy, while finding an upper-bound path is generally hard.

For some types of graphs, it has been shown that burning them in the fewest possible number of steps is generally hard to do. One such type of graphs are path-forests, which are composed of paths that are completely separate from each other. We use this to prove that cycle-forests, which are similar to path-forests, are also hard to burn in an optimal way.

1 Introduction

In recent years, the study of networks has become increasingly important. For instance, social media has grown to play a vital role in today's society as a source of information and a means to communicate.

One way to study a network mathematically is by representing it as a graph. A graph is a collection of vertices and edges in which each vertex illustrates an object from the network and each edge depicts a connection between two vertices. For instance, vertices may be interpreted as individuals on a social media platform and edges as interactions between these individuals. An example of a graph is shown in Figure 1.1.

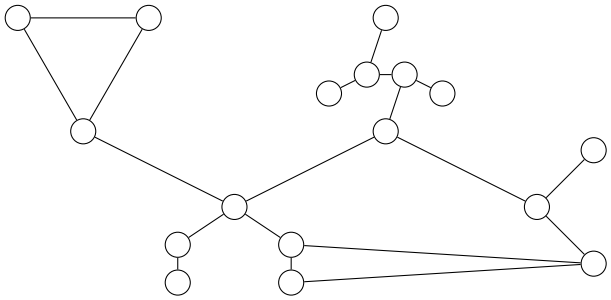
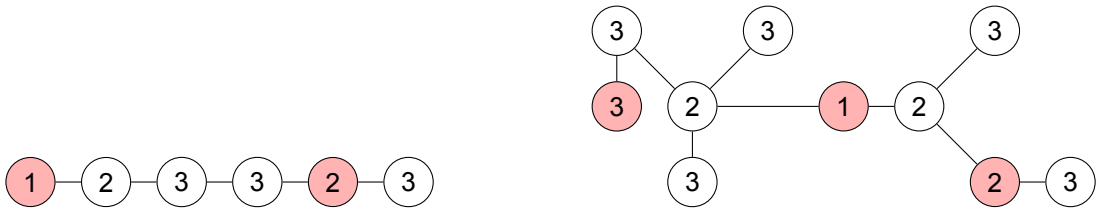


Figure 1.1: A graph in which the circles are vertices and the lines between circles are edges.

To examine how quickly information spreads across a network, a model was introduced in [3]. In this model, which considers the process of *graph burning*, the concept of a spreading fire was translated to graphs. The process takes place in steps (or rounds) and is as follows. At first, all vertices are in an 'unburned' state. Then, in round 1, a single vertex is chosen to be the first source of fire. This vertex is now burned. In round 2, the fire spreads from the first source to all its neighbors. That is, all vertices that are directly connected to the first source become burned. In addition, a second source of fire may be chosen which also burns immediately. Each round, the fire spreads from burned vertices to their unburned neighbors and an additional source of fire may be selected. This process continues until all vertices of the graph are burned. Examples of the graph burning process are shown in Figure 1.2.



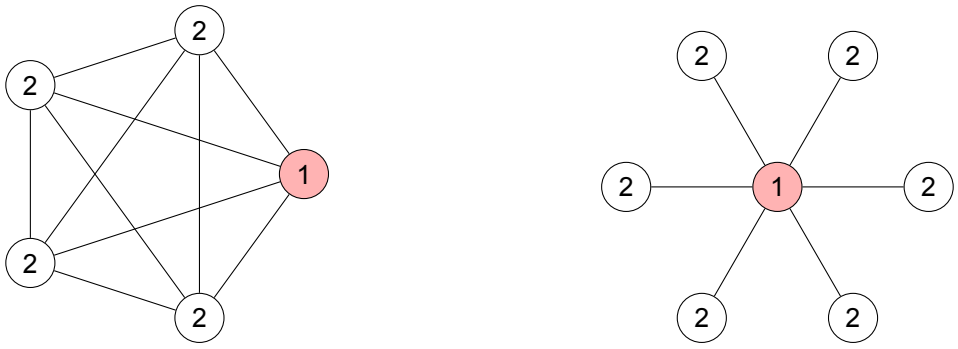
(a) Burning a path graph in 3 rounds using 2 sources.

(b) Burning a tree in 3 rounds using 3 sources.

Figure 1.2: Examples of graph burning; sources are marked in red and the numbers in vertices indicate the round in which they are burned.

Choosing different vertices to be sources may affect the number of rounds needed to fully burn a graph. The aim is to find the minimum number of rounds. This number is known as the *burning number* of the graph. For some classes of graphs, the burning number can be quickly identified. One such class is the class of *complete graphs*, which are graphs where each vertex is connected to all other vertices. Complete graphs have burning number 2, as any first source burns all other vertices in the second round. A similar argument is true for *star graphs*: star graphs have exactly one vertex that is connected to vertices. These examples are illustrated in Figure 1.3.

Although the burning number has been determined for some graph families, finding it for arbitrary graphs remains an unsolved problem. Two classes of graphs that play a pivotal role in making progress in this problem are *path graphs* and *trees* (see Figure 1.2 for examples). For path graphs, the burning number is known and it is conjectured to be the highest burning number of all graphs (Conjecture 2.1). If all trees are proven to satisfy this conjecture, then all graphs do. Path graphs and trees will be formally introduced in Chapter 2; their role in graph burning will be discussed in more detail in Chapter 3.



(a) Complete graphs have burning number 2.

(b) Star graphs have burning number 2.

Figure 1.3: Examples of graphs with burning number 2.

Different approaches have been taken to study graph burning, such as finding bounds for the burning number and developing algorithms that produce burning sequences for graphs. Before some of the important results are discussed, in Chapter 2 we will focus on setting up the mathematical framework to formalize graph burning. In Chapter 3, we will explore previously established results in graph burning, which can be primarily classified into two categories: results directly related to finding the burning number (section 3.1) or algorithms aimed at burning graphs as efficiently as possible (section 3.2).

Chapters 4 and 5 are dedicated to our contributions. In Chapter 4, we will construct *necklace graphs* and find bounds on their burning number. They are constructed by stringing together smaller graphs (*pearls*) which may be heavily interconnected. The goal of this construction is to obtain a graph that burns as slowly as possible, due to its path-like structure. In Chapter 5, we will consider a specific class of graphs, called *cycle-forests* (see definition in Chapter 2). We will prove that the graph burning problem is NP-complete for this class of graphs. Informally, this means that finding an optimal burning schedule for cycle-forests is computationally hard.

2

Preliminaries

In this chapter, we will review fundamental definitions from graph theory (section 2.1) and describe the process of graph burning in a formal way (section 2.2).

2.1. Concepts and definitions from graph theory

We start by introducing basic concepts and definitions from graph theory, which will lay the foundation for graph burning. A *graph* is a pair of sets $G = (V, E)$, where the elements of V (or $V(G)$) are called *vertices* and the elements of E (or $E(G)$) are unordered pairs of vertices, also known as *edges*. The number of vertices in a graph G, also known as the *order* of G, is written as $|G|$. A graph is called *simple* if it does not contain edges from a vertex to itself (*loops*) and if between any two vertices there is at most one edge connecting the two. If two vertices u, v are connected by an edge, we denote this edge by uv and call u and v *adjacent* or *neighbors*. We write the set of neighbors of v as $N(v)$. We say that the *endpoints* u and v of uv are *incident* to uv. The *degree* $d(v)$ of a vertex v is the number of edges it is incident to (or equivalently, the number of neighbors of v). If all vertices in a graph have the same degree k, the graph is said to be *k-regular*. A graph in which each vertex is connected to every other vertex is called *complete*, and it is denoted by K_n, where n is the order of the graph. A *subdivision* of an edge uv is the action of removing uv and adding a vertex w, along with the edges uw and wv. A *subgraph* of $G = (V, E)$ is a graph $G' = (V', E')$ such that $V' \subseteq V$ and $E' \subseteq E$. An *induced* subgraph of $G = (V, E)$ on a subset $V' \subseteq V$ contains exactly the vertices in V' and all of the edges between these vertices that are present in the original graph G. In other words, the edge set E' of G' satisfies $E' = \{uv \in E : u, v \in V'\}$. We denote a subgraph $G' \subseteq G$ induced by $V' \subseteq V$ as $G[V']$.

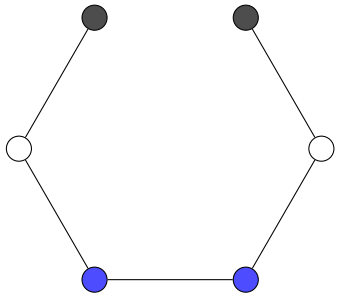
Given any vertex u in a graph G, one might wonder whether it is possible to reach another vertex v by traversing edges of G. This gives rise to the following definition. A *path* is a sequence of distinct vertices $u = v_1, v_2, \dots, v_{n-1}, v_n = v$ from u to v such that $v_i v_{i+1} \in E$ for $i \in \{1, \dots, n-1\}$. A graph is called *connected* if there exists a path between any two vertices. A disconnected graph whose connected subgraphs are not part of a larger subgraph are called *components*. If removing a vertex (along with its incident edges) from a graph would increase the number of connected components in a graph, it is said to be a *cut vertex*. A *k-connected* graph is a graph that remains connected whenever fewer than k vertices are removed from it. These notions regarding the connectivity of a graph are illustrated in Figure 2.1.

(a) A disconnected graph (path-forest) with 3 components.

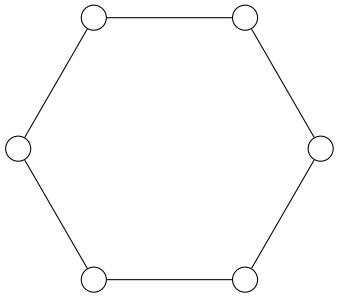
(b) A 2-connected graph.

Figure 2.1: Examples of a disconnected and a 2-connected graph.

The *distance* between two vertices u and v, denoted by $d(u,v)$, is the *length* of a shortest path between u and v (that is, the number of edges in the path). The maximum distance from a certain vertex u to any other vertex in the graph is the *eccentricity* of u: $\mathrm{ecc}(u) = \max\{d(u,v) : v \in V(G)\}$. The minimum eccentricity in a graph is called the *radius* of the graph, while the maximum eccentricity is called the *diameter*. They are denoted as $\mathrm{rad}(G)$ and $\mathrm{diam}(G)$, respectively. Figure 2.2 demonstrates the difference between radius and diameter:



(a) A path graph with radius 3 and diameter 5; blue vertices are associated with the radius, black vertices with the diameter.

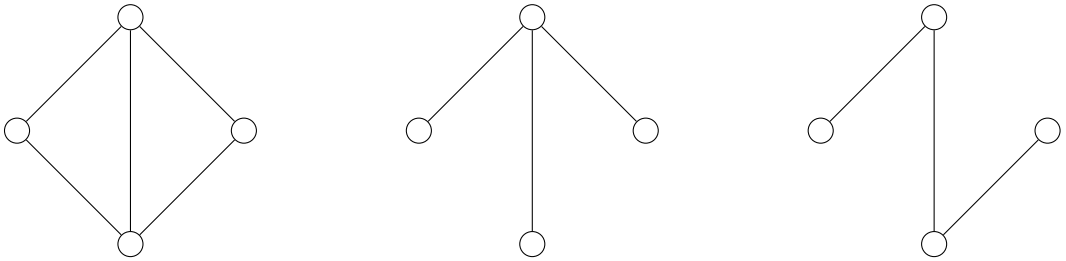


(b) A cycle graph with radius 3 and diameter 3; every vertex has both the radius and diameter as eccentricity.

Figure 2.2: Examples of the radius and diameter of graphs.

A graph whose vertices can be ordered as $v_1, v_2, \ldots, v_{n-1}, v_n$ such that the edges are $v_i v_{i+1}$ for $i \in \{1, \ldots, n-1\}$ is a *path graph*, written as P_n. A cycle is a path whose starting and ending point are the same. A *cycle graph* on n vertices is a graph that consists of a single cycle, and it is denoted by C_n. Path graphs and cycle graphs play an important role in graph burning.

Another important class of graphs are *trees*. Trees are connected graphs that contain no cycles. Every tree has at least one *leaf*, which is a vertex of degree 1. A *forest* is a graph whose connected components are trees. In particular, a *path-forest* is a graph whose components are paths (see Figure 2.1). Similarly, a *cycle-forest* is a disconnected graph composed of cycles. Strictly speaking, a cycle-forest is not a forest as its connected components are not trees. However, it will be convenient to refer to cycle-forests in this way to emphasize their similarity to path-forests. Cycle-forests will be discussed in more detail in Chapter 5. A *spanning tree* is a subgraph $T \subseteq G$ which is a tree and contains all vertices of G. Figure 2.3 shows that a graph may have multiple spanning trees.



(a) A graph (which is not a tree). (b) Spanning tree 1. (c) Spanning tree 2.

Figure 2.3: A graph with two spanning trees.

For further background information on graph theory, we refer the reader to [7].

2.2. Graph burning

Having introduced basic definitions in graph theory, we are now set to formalize graph burning. We start with a finite, simple and connected graph G. The process of graph burning is a discrete process (a process divided into rounds) where G is burned according to the following rules. Let $S = (x_1, x_2, \ldots, x_k)$ be a sequence of vertices, which are called *sources*. Initially, in round 0, all vertices are unburned. In round 1, we burn the source x_1. For each round i that follows ($i \in \{2, \ldots, k\}$), we burn the source x_i and all unburned neighbors of burned vertices. After k rounds, if all vertices of G are burned, S is called a *burning sequence*. The *burning number* $b(G)$ is the minimum number of rounds needed to burn G. In

[3], it was shown that $b(P_n) = \lceil \sqrt{n} \rceil$. It was conjectured that among all connected graphs, path graphs have the highest burning number.

Conjecture 2.1 (Burning Number Conjecture [3])**.** *Let G be a connected graph on n vertices. Then* $b(G) \leq \lceil \sqrt{n} \rceil$.

Although Conjecture 2.1 still remains unproven in general, it has been verified for various specific cases. These will be discussed in more detail in Chapter 3.

3

Literature review

In this chapter, we collect some important results that have been established in graph burning. We start by discussing graph classes that have been shown to satisfy Conjecture 2.1 or whose burning number has been identified (section 3.1). Finding the burning number of a graph is NP-complete [2], which has led to the development of (approximation) algorithms for the burning number of a graph and complexity results (section 3.2).

3.1. Burning number

A line of research has focused on identifying or finding bounds for the burning number for different classes of graphs. In this section, we will explore these results.

We begin by stating a characterization of the burning number of graphs in terms of trees, which was originally given in [3].

Corollary 3.1 ([3]). *For a graph G, we have that*

$$b(G) = \min\{b(T) : T \text{ is a spanning subtree of } G\}.$$

It follows from this result that Conjecture 2.1 is true for all connected graphs if it is true for trees. Some families of trees have already been shown to satisfy Conjecture 2.1. The burning number of path graphs was determined in [3].

Proposition 3.1 ([3]). *For a path graph P_n on n vertices, we have that $b(P_n) = \lceil \sqrt{n} \rceil$.*

In [5], Conjecture 2.1 was shown to be true for spider graphs, which are trees with exactly one vertex of degree greater than or equal to 3.

Proposition 3.2 ([5]). *Let G be a spider graph on n vertices. Then $b(G) \leq \lceil \sqrt{n} \rceil$.*

In [13], graph burning was studied on p-caterpillars. A p-caterpillar G is a tree whose vertices are within a distance p of a so-called 'central spine', which is the longest path in G. A caterpillar is short for a 1-caterpillar. Furthermore, given a p-caterpillar G, a p-leg is a subgraph of G whose vertices lie on a path of length p and are not on the central spine. The following results were derived.

Proposition 3.3 ([13]). *Let G be a caterpillar on n vertices. Then $b(G) \leq \lceil \sqrt{n} \rceil$.*

Proposition 3.4 ([13]). *Let G be a 2-caterpillar on n vertices. Then $b(G) \leq \lceil \sqrt{n} \rceil$.*

Proposition 3.5 ([13]). *Let G be a 3-caterpillar with at least $2\lceil \sqrt{n} \rceil - 1$ leaves. Then $b(G) \leq \lceil \sqrt{n} \rceil$.*

Proposition 3.6 ([13]). *Let G be a p-caterpillar with at least $2\lceil \sqrt{n} \rceil - 1$ disjoint p-legs. Then $b(G) \leq \lceil \sqrt{n} \rceil$.*

Conjecture 2.1 has also been verified for homeomorphically irreducible trees (HITs), which are trees without vertices of degree 2 [18].

Proposition 3.7 ([18]). *Let G be a HIT on n vertices. Then $b(G) \leq \lceil \sqrt{n} \rceil$.*

Note that by Corollary 3.1, the Burning Number Conjecture is true for any graph that contains one of the trees listed above as a spanning tree. For example, cycle graphs C_n have burning number $b(C_n) = \lceil \sqrt{n} \rceil$ as they only have paths as spanning trees.

In addition to trees, the burning number has been studied for various other classes of graphs. In [9], the burning number of 3-regular and several other specific circulant graphs was determined, and bounds were given for the burning number of 4-regular circulant graphs. In [19], Conjecture 2.1 was verified asymptotically for generalized Petersen graphs and also proven for specific generalized Petersen graphs. Various bounds for the burning numbers of graph products were provided in [17]. A lower bound for the burning number of grids of size $l \times b$ was given in [12]: it is at least $(l \times b)^{\frac{1}{3}}$. Specific types of theta graphs were studied in [16] and their burning number was given. In [1], an improved best-known bound on the burning number of a graph G was given: $b(G) \leq \left\lceil \sqrt{\frac{4}{3}n} \right\rceil + 1$. This bound was used to show that Conjecture 2.1 is almost valid for graphs of minimum degree 3 ($b(G) \leq \lceil \sqrt{n} \rceil + 2$) and is valid for large enough graphs of minimum degree 4, which are graphs whose vertices have degree at least 3 or 4, respectively. Finally, graphs with burning number 3 were characterized in [15].

3.2. Algorithms and complexity results

To make an attempt at finding the burning number of an arbitrary graph, one possible approach is the development of an algorithm. Such an algorithm is a sequence of instructions that takes a graph as input and returns its burning number as output. For the algorithm to work efficiently in practice, it should return an optimal solution within a reasonable amount of time. This is where difficulties arise: finding the burning number of a graph is an NP-complete problem, even for graph families with a relatively simple structure, such as spider graphs and path-forests [2]. NP is the class of decision problems where ”yes” instances (the problems that have a solution) have a proof (or certificate) that can be verified quickly, even if finding those certificates might be hard. In the context of NP, ”quickly” means ”in polynomial time”, which describes that the time needed to check a certificate grows at a rate proportional to a polynomial function of the input size. If a certificate can also be found in polynomial time, the decision problem is in the class P. Unless each decision problem verifiable in polynomial time is also solvable in polynomial time (that is, P = NP), there is no algorithm that can find the burning number of a graph in polynomial time. A decision problem is NP-hard if every problem in NP can be reduced to it in polynomial time. If a decision problem is in NP and NP-hard, it is called NP-complete. Therefore, the graph burning problem is at least as hard as other decision problems in NP. This has raised the importance of approximation algorithms and heuristics. Approximation algorithms are algorithms that run in polynomial time and return a solution that is guaranteed to be worse than the optimal solution only by a constant factor. Heuristic algorithms usually run in polynomial time as well, but they do not give a guarantee on the quality of the produced solution. In this section, we will discuss some (approximation) algorithms and heuristics that are currently known to find or approximate the burning number of a graph. Moreover, we will list some complexity results regarding the graph burning decision problem.

Burning spider graphs and path-forests was shown to be an NP-complete problem in [2]. However, polynomial-time algorithms finding their burning numbers were provided under certain restrictions on the number of components or orders. Also, a polynomial time 3-approximation algorithm was given for general graphs. More progress has been made on burning path-forests. In [5], a $\frac{3}{2}$-approximation algorithm was introduced for the burning number of path-forests. If a path-forest consists of a constant number of disjoint paths, its burning number can be found in polynomial time [4]. In the same paper, two polynomial-time approximation algorithms were given that approximate the burning number of path-forests with a non-constant number of components. Moreover, polynomial-time approximation algorithms were given that approximate the burning number with approximation ratio 3 for general graphs and 2 for trees. In [14], an algorithm was presented that burns any graph with minimum degree δ in at most $\left\lceil \sqrt{\frac{24n}{\delta+1}} \right\rceil$ rounds. It follows that Conjecture 2.1 is true for graphs with minimum degree $\delta \geq 23$. A 2-approximation algorithm for burning square grids was provided in [12]. Moreover, it was shown that burning connected interval graphs and permutation graphs is NP-complete.

In addition to the development of algorithms for specific graph families, heuristics and algorithms have been designed aimed at burning graphs from commonly used datasets, which are more representative of real-life networks. The first heuristics for graph burning were introduced in [8], where they

were tested both on known datasets for other NP-hard problems and randomly generated graphs whose burning number is known. A key aspect of these heuristics is 'centrality' of the first source: the first source is chosen in such a way that the number of vertices of G burned after $b(G)$ rounds is maximized. Another aspect that is taken into account is trying to burn those vertices sooner which are farthest away from previous sources. A different approach was taken in [10]: rather than burning graphs based on centrality measures, this heuristic burns graphs 'greedily'. That is, each round a source is chosen such that it covers as many unburned vertices as possible. The best up-to-date algorithm was given in [6]: it optimally burns graphs with up to 200,000 vertices in under 35 seconds.

4

Necklace graphs

In this chapter, we will introduce a new class of graphs which we call necklace graphs (or necklaces for short). We will find upper and lower bounds for the burning number $b(G)$ of any member G of this class.

We begin by defining pearls, which are the building blocks of necklaces.

Definition 4.1 (Pearl)**.** *Let G be a finite, simple, and connected graph. We call G with designated vertices s, t a pearl.*

To construct a lower bound for the burning number of a pearl, we define *lower-bound paths*.

Definition 4.2 (Lower-bound path)**.** *Let $G = (V, E)$ be a pearl with designated vertices s, t. We call $P \subseteq G$ a lower-bound path if it is a path from s to t such that*

$$\forall v \in G \setminus P \quad \exists p \in P \quad \forall p' \in P : \quad vp' \in E \Rightarrow pp' \in E.$$

Every pearl contains a lower-bound path. The following theorem demonstrates that a shortest path between the designated vertices of the pearl is always a lower-bound path.

Theorem 4.1. *Let G be a pearl with designated vertices s, t. Then a shortest s-t path P is a lower-bound path for G.*

Proof. To show that a shortest path P from s to t is a lower-bound path, we need to verify that it satisfies the definition of a lower-bound path. For every vertex $v \in G \setminus P$, there must exist a vertex $p \in P$ such that p is adjacent to all vertices $p' \in P$ which are neighbors of v. That is, v should satisfy the condition $N(v) \cap P \subseteq N(p) \cap P$. Several cases may arise regarding the number of vertices in $N(v) \cap P$.

Case 1: v has 0 neighbors in P. In this case, the condition is satisfied immediately ($N(v) \cap P = \emptyset$).

Case 2: v has 1 neighbor in P, say p'_1. We can assign p'_1 the role of p in the definition.

Case 3: v has 2 neighbors in P, say p'_1, p'_3. We can identify either of them with p if $p'_1p'_3 \in E(P)$; otherwise, there exists exactly one vertex $p'_2 \in P$ such that $p'_1p'_2, p'_2p'_3 \in E(P)$, and we identify p'_2 with p. Indeed, if there were $k > 1$ vertices between p'_1 and p'_3 on the path, say $p'_{2,1}, p'_{2,2}, \dots, p'_{2,k}$, then replacing the sequence $s, \dots, p'_1, p'_{2,1}, \dots, p'_{2,k}, p'_3, \dots, t$ in P by the sequence $s, \dots, p'_1, v, p'_3, \dots, t$ would yield a shorter path from s to t, which is a contradiction.

Case 4: v has 3 neighbors in P, say p'_1, p'_2, p'_3. These three vertices must be consecutive vertices in P. If they were not, so $p'_1p'_2 \notin E(P)$ or $p'_2p'_3 \notin E(P)$, then we arrive at the same contradiction as in the third case. Therefore, we can assign p'_2 the role of p in the definition.

To see that v can have no more than 3 neighbors in P, we assume that it has neighbors p'_1, p'_2, p'_3, p'_4 which are consecutive vertices in P, and we derive a contradiction. The sequence $s, \dots, p'_1, v, p'_4, \dots, t$ is a shorter sequence of vertices in P than the sequence $s, \dots, p'_1, p'_2, p'_3, p'_4, \dots, t$. This is a contradiction, since we assumed P with the latter sequence of vertices to be a shortest s-t path.

From the case analysis, it follows that every $v \in G \setminus P$ satisfies the lower-bound path condition. We may thus conclude that a shortest s-t path is always a lower-bound path. □

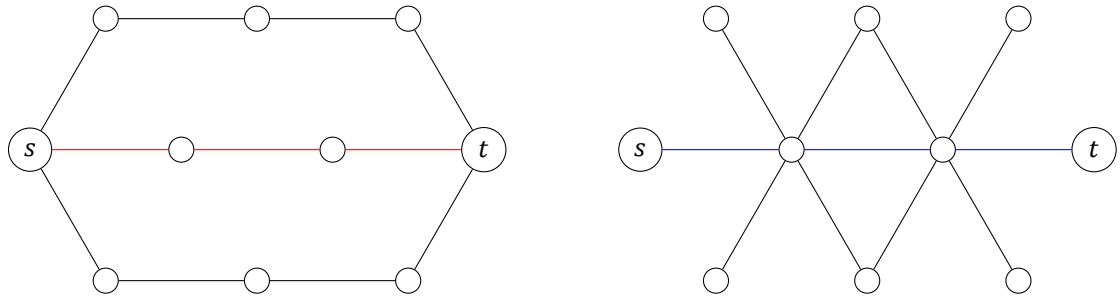
Since a shortest path between two vertices in a graph can be found in polynomial time, if follows from Theorem 4.1 that we can find a lower-bound path in polynomial time.

We now define *upper-bound paths*, which will provide an upper bound for the burning number of a pearl.

Definition 4.3 (Upper-bound path)**.** *Let $G = (V, E)$ with designated vertices s, t be a pearl. We call $P \subseteq G$ an upper-bound path if it is a path from s to t such that*

$$\forall v \in G \setminus P \quad \exists p \in P \setminus \{s, t\} : \quad vp \in E.$$

See Figure 4.1 for examples of a lower-bound path and an upper-bound path.



(a) A lower-bound path in a pearl, highlighted in red. Note that this is a shortest s-t path.

(b) An upper-bound path in a pearl, highlighted in blue. Note that this is also a lower-bound path.

Figure 4.1: Examples of a lower-bound path and an upper-bound path in two pearls.

Unlike lower-bound paths, upper-bound paths do not always exist. In fact, it is computationally hard to find an upper-bound path for an arbitrary pearl. We will prove in Theorem 4.2 that finding an upper-bound path in an arbitrary pearl is NP-complete. First, we formulate the Upper-Bound Path decision problem.

Problem: Upper-Bound Path
Input: A pearl G with two distinct vertices s, t and an integer k.
Question: Does there exist an upper-bound path in G of length at most k?

We will prove that the Upper-Bound Path problem is NP-complete by making a reduction from the following known NP-complete problem [11]:

Problem: Hamiltonian Path
Input: A pearl G with two distinct vertices s, t.
Question: Does there exist a Hamiltonian path in G that starts at s and ends at t?

Theorem 4.2. *The Upper-Bound Path problem is NP-complete.*

Proof. We begin by verifying that the Upper-Bound Path problem is in NP. Suppose we are given a pearl G with a candidate path P from s to t of length at most k. We can verify that P is a path, check its length, and for each vertex $v \in G \setminus P$ confirm that it has a neighbor in P. All of these can be done in polynomial time. Thus, the problem is in NP.

Now, let $G = (V, E)$ be an instance of the Hamiltonian Path problem with $|V| = n$ and designated endpoints s and t. We construct a new graph $G' = (V', E')$ as follows. For every edge $uv \in E$, subdivide it by introducing a new vertex w_{uv}, and replace uv with the edges uw_{uv} and $w_{uv}v$. Let $W = \{w_{uv} : uv \in E\}$ be the set of subdivision vertices. Set $V' = V \cup W$ and $E' = \{uw_{uv}, w_{uv}v : uv \in E\}$. Finally, set $k := 2n - 1$.

We now show that G has a Hamiltonian path from s to t if and only if G' has an upper-bound path from s to t of length at most k.

Suppose G has a Hamiltonian path from s to t, which we denote by $P : s = v_1, v_2, \dots, v_{n-1}, v_n = t$. We define a path P' in G' as follows: $P' : s = v_1, w_{v_1v_2}, v_2, \dots, v_{n-1}, w_{v_{n-1}v_n}, v_n = t$. This is indeed possible, since for all $i \in \{1, \dots, n-1\}$, we have $w_{v_iv_{i+1}} \in V'$ and $v_iw_{v_iv_{i+1}}, w_{v_iv_{i+1}}v_{i+1} \in E'$. Furthermore, P' has

length $2n-1=k$ as it consists of all n vertices from V and all $n-1$ subdivision vertices from W. In particular, $V' \setminus V(P') = \emptyset$. Thus, P' is an upper-bound path for G', as there are no vertices outside P'.

Conversely, suppose G' has an upper-bound path P' from s to t of length at most k. Note that every vertex on P' is either in V or in W. Suppose, for contradiction, that there is some vertex $v \in V$ which is not on P'. Every vertex in V is adjacent only to subdivision vertices in W; similarly, every subdivision vertex $w_{uv} \in W$ is adjacent only to $u \in V$ and $v \in V$. Therefore, if w_{uv} is on P', then both u and v must also be on P'. So if any vertex $v \in V$ is not on the path P', then none of its neighboring subdivision vertices can be on P' either. But then v has no neighbor on P', contradicting the fact that P' is an upper-bound path. Hence, every vertex $v \in V$ must be on P'. Since every pair of vertices in V is separated by a subdivision vertex, P' must alternate between vertices in V and vertices in W. That is, we can order the vertices of P' as $s = v_1, w_{v_1v_2}, v_2, \dots, v_{n-1}, w_{v_{n-1}v_n}, v_n = t$, where $v_j \in V$, $w_{v_iv_{i+1}} \in W$ for $1 \le i \le n-1$ and $1 \le j \le n$. Thus, $P\colon\ s = v_1, v_2, \dots, v_{n-1}, v_n = t$ is a Hamiltonian path in G, since each $w_{v_iv_{i+1}} \in W$ corresponds to the edge v_iv_{i+1} in E.

It follows from the reduction that the Upper-Bound Path problem is NP-hard. Since the problem is also in NP, we conclude that the Upper-Bound Path problem is NP-complete. □

Having defined a lower-bound path and an upper-bound path for pearls, we can now extend these concepts to the main class of graphs of this chapter: necklaces.

Definition 4.4 (Necklace)**.** *Let $G = (V, E)$ be a finite, simple and connected graph. We call G a necklace if for some $N \in \mathbb{N}$ and for every $i \in \{1, 2, \dots, N\}$, there exist disjoint pearls $G_i \subseteq G$ with designated vertices s_i, t_i such that $V(G) = \bigcup_{i=1}^{N} V(G_i)$ and $E(G) = \bigcup_{i=1}^{N} E(G_i) \cup \{t_j s_{j+1} : 1 \le j \le N-1\}$. The designated vertices of G are s_1, t_N.*

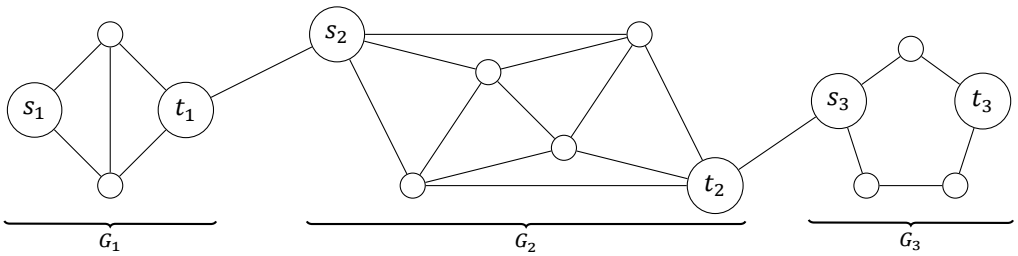


Figure 4.2: Example of a necklace with pearls G_1, G_2 and G_3. The labeled vertices are designated vertices.

Note that by definition, necklaces are pearls. An example of a necklace is given in Figure 4.2. Unless specified otherwise, we assume that a necklace G consists of pearls $G_1, \dots, G_N$ where each pearl G_i has designated vertices s_i, t_i. The designated vertices of G are s_1 and t_N. We aim to generalize the notions of a lower-bound path and an upper-bound path for pearls to necklaces. The following lemma shows how a lower-bound path for G can be constructed from lower-bound paths for its pearls.

Lemma 4.1. *Let G be a necklace and suppose that each of its pearls G_i has a lower-bound path $P_{l,i}$. Define P_l to be the path such that $V(P_l) = \bigcup_{i=1}^{N} V(P_{l,i})$ and $E(P_l) = \bigcup_{i=1}^{N} E(P_{l,i}) \cup \{t_j s_{j+1} : 1 \le j \le N-1\}$. Then P_l is a lower-bound path for G.*

Proof. To show that P_l is a lower-bound path for G, we need to verify that $\forall v \in G \setminus P_l\ \exists p \in P_l\ \forall p' \in P_l\colon\ vp' \in E \Rightarrow pp' \in E$. So let $v \in G \setminus P_l$ be arbitrary. Note that $v \in G_i \setminus P_{l,i}$ for some $i \in \{1, \dots, N\}$. Since $P_{l,i}$ is a lower-bound path for G_i, we know that there exists $p_i \in P_{l,i}$ such that for every $p' \in P_{l,i}$, if $vp' \in E(G_i)$, then $p_ip' \in E(G_i)$. Since the only possible neighbors of v in P_l are in $P_{l,i}$, we can choose $p := p_i$ and it follows that $\forall p' \in P_l\colon\ vp' \in E \Rightarrow pp' \in E$. Thus, P_l is a lower-bound path for G. □

Using a similar argument, we can construct an upper-bound path P_u for G given that each pearl G_i has an upper-bound path $P_{u,i}$. This is formalized in the following lemma.

Lemma 4.2. *Let G be a necklace and suppose that each of its pearls G_i has an upper-bound path $P_{u,i}$. Define P_u to be the path such that $V(P_u) = \bigcup_{i=1}^{N} V(P_{u,i})$ and $E(P_u) = \bigcup_{i=1}^{N} E(P_{u,i}) \cup \{t_j s_{j+1} : 1 \le j \le N-1\}$. Then P_u is an upper-bound path for G.*

Proof. To show that P_u is an upper-bound path for G, we need to verify that $\forall v \in G \setminus P_u \; \exists p \in P_u : vp \in E$. So let $v \in G \setminus P_u$ be arbitrary. Note that $v \in G_i \setminus P_{u,i}$ for some $i \in \{1, \dots, N\}$. Since $P_{u,i}$ is an upper-bound path for G_i, we know that there exists $p_i \in P_{u,i}$ such that $vp \in E(G_i)$. Therefore, choose $p := p_i$ and it follows that $vp \in E$. Thus, P_u is an upper-bound path for G. □

In Theorem 4.3 and 4.4, we provide a lower bound and an upper bound for the burning number of pearls.

Theorem 4.3. *If G is a pearl with lower-bound path P_l, then $b(G) \geq b(G[V(P_l)])$.*

Proof. Let G be a pearl with burning sequence $S_G = (x_1, x_2, \dots, x_k)$. Since G has a lower-bound path P_l, we know that for any $v \in G \setminus P_l$, there exists a $p \in P_l$ such that for all $p' \in P_l$, $vp' \in E(G)$ implies $pp' \in E(G)$. We construct a sequence $S_{P_l} = (y_1, y_2, \dots, y_k)$ as follows. For every $x_j \notin P_l$ $(1 \leq j \leq k)$, choose $y_j \in P_l$ such that for all $p' \in P_l$, $y_j p' \in E(G)$ whenever $x_j p' \in E(G)$. For $x_j \in P_l$, take $y_j = x_j$. We claim that S_{P_l} is a burning sequence for $G[V(P_l)]$, the subgraph of G induced by the vertices of P_l. If this claim holds, we have constructed a burning sequence S_{P_l} from S_G of equal length. It then follows that $b(G) \geq b(G[V(P_l)])$. To verify the claim, it suffices to show that for each $y_j \neq x_j$ the vertices of P_l are burned by y_j earlier than or in the same round as by x_j. In fact, we only need to consider neighbors of x_j on P_l: if y_j burns those in the same round as x_j, then it burns all vertices on P_l in (at most) the same round as x_j. But by assumption, each vertex of P_l which is a neighbor of x_j is also a neighbor of y_j (that is, $N(x_j) \cap P_l \subseteq N(y_j) \cap P_l$). This means that each neighbor of x_j is burned by y_j in the same round as by x_j (or earlier), and so the claim follows. We may thus conclude that $b(G) \geq b(G[V(P_l)])$. □

Theorem 4.4. *If G is a pearl with an upper-bound path P_u, then $b(G) \leq b(P_u) + 1$.*

Proof. Let G be a pearl with an upper-bound path P_u. By definition of an upper-bound path, we have that for every $v \in G \setminus P_u$, there exists a $p \in P_u$ such that $vp \in E(G)$. Therefore, v is burned at most 1 round after its neighbor on P_u. Since all vertices on P_u can be burned in at most $b(P_u)$ rounds, it follows that v burns after at most $b(P_u) + 1$ rounds. Thus, $b(G) \leq b(P_u) + 1$. □

Building on the previously derived results, we now present the main result concerning the burning number of necklaces.

Theorem 4.5. *Let G be a necklace with pearls $G_1, G_2, \dots, G_N$. Suppose that each pearl G_i admits a lower-bound path $P_{l,i}$ and an upper-bound path $P_{u,i}$. Then $b(G[\bigcup_{i=1}^{N} V(P_{l,i})]) \leq b(G) \leq \left\lceil \sqrt{\sum_{i=1}^{N} |P_{u,i}|} \right\rceil + 1$.*

Proof. By Lemma 4.1, we can construct a lower-bound path P_l for G, which contains the vertices of each $P_{l,i}$. Therefore, we have $V(P_l) = \bigcup_{i=1}^{N} V(P_{l,i})$. Theorem 4.3 now implies that $b(G) \geq b(G[V(P_l)]) = b(G[\bigcup_{i=1}^{N} V(P_{l,i})])$, which establishes the first inequality.

Lemma 4.2 provides us with an upper-bound path P_u such that $|P_u| = \sum_{i=1}^{N} |P_{u,i}|$. It follows from Theorem 4.4 and Proposition 3.1 that $b(G) \leq b(P_u) + 1 = \lceil \sqrt{|P_u|} \rceil + 1 = \left\lceil \sqrt{\sum_{i=1}^{N} |P_{u,i}|} \right\rceil + 1$, which verifies the second inequality. □

The following definition classifies pearls that satisfy an additional property compared to general pearls. This property can be used to reduce the number of values that the burning number of a corresponding necklace can take, which is formulated in Corollary 4.1.

Definition 4.5 (Perfect pearl)**.** *Let $G = (V, E)$ be a pearl with designated vertices s, t. We say G is perfect if it contains a lower-bound path P_l and an upper-bound path P_u such that $|P_l| = |P_u|$.*

Corollary 4.1. *Let G be a necklace with perfect pearls $G_1, G_2, \dots, G_N$. Suppose that each pearl G_i admits a lower-bound path $P_{l,i}$ and upper-bound path $P_{u,i}$. Then $b(G[\bigcup_{i=1}^{N} V(P_{l,i})]) \leq b(G) \leq \left\lceil \sqrt{\sum_{i=1}^{N} |P_{l,i}|} \right\rceil + 1$.*

Proof. Since each pearl G_i is perfect, we have $|P_{l,i}| = |P_{u,i}|$. By Theorem 4.5, G then satisfies $b(G[\bigcup_{i=1}^{N} V(P_{l,i})]) \leq b(G) \leq \left\lceil \sqrt{\sum_{i=1}^{N} |P_{l,i}|} \right\rceil + 1$. □

To see how Corollary 4.1 can improve the estimation of the burning number of a necklace, we consider pearls with designated vertices s, t such that a shortest s-t path is an upper-bound path. This leads to the following result.

Corollary 4.2. *Let G be a necklace with pearls $G_1, G_2, \dots, G_N$. Suppose that for each pearl G_i, a shortest s_i-t_i path P_i is an upper-bound path. Then either $b(G) = \left\lceil \sqrt{\sum_{i=1}^{N} |P_i|} \right\rceil$ or $b(G) = \left\lceil \sqrt{\sum_{i=1}^{N} |P_i|} \right\rceil + 1$.*

Proof. Theorem 4.1 implies P_i is a lower-bound path for G_i, and by assumption P_i is also an upper-bound path for G_i. By Lemma 4.1 and 4.2, the path P containing the vertices of each P_i is both a lower-bound path and an upper-bound path for G. Since P_i is a shortest s_i-t_i path, we have $b(G[V(P_i)]) = b(P_i)$. Similarly, P is a shortest path from s_1 to t_N, and so $b(G[\bigcup_{i=1}^{N} V(P_i)]) = b(P)$. By Proposition 3.1, $b(P) = \lceil \sqrt{|P|} \rceil = \left\lceil \sqrt{\sum_{i=1}^{N} |P_i|} \right\rceil$. It follows from Corollary 4.1 that $\left\lceil \sqrt{\sum_{i=1}^{N} |P_i|} \right\rceil \leq b(G) \leq \left\lceil \sqrt{\sum_{i=1}^{N} |P_i|} \right\rceil + 1$. That is, either $b(G) = \left\lceil \sqrt{\sum_{i=1}^{N} |P_i|} \right\rceil$ or $b(G) = \left\lceil \sqrt{\sum_{i=1}^{N} |P_i|} \right\rceil + 1$. □

For necklaces where a shortest path is an upper-bound path, Corollary 4.2 reduces the number of values the burning number can take to 2.

An example of such a necklace is the necklace G whose pearls $G_1, G_2, \dots, G_N$ are complete graphs: $G_i = K_{n_i}$ with $n_i = |G_i|$, $i \in \{1, \dots, N\}$. Define the path P in G as the path containing the designated vertices of each pearl G_i: $P: s_1, t_1, s_2, \dots, t_{N-1}, s_N, t_N$. Note that P contains $2N$ vertices. P is a shortest s_1-t_N path, so by Theorem 4.1, P is a lower-bound path. Moreover, P is an upper-bound path: for all $v \in G_i \setminus \{s_i, t_i\}$, we have $s_i v \in E(G)$ as s_i is connected to all other vertices in G_i. It follows from Corollary 4.2 that either $b(G) = \lceil \sqrt{2N} \rceil$ or $b(G) = \lceil \sqrt{2N} \rceil + 1$.

5

Cycle-forests

In this chapter, we prove that the Graph Burning problem is NP-complete for cycle-forests, which were defined in Chapter 2. We do this by making a reduction from the Graph Burning problem for path-forests, which was shown to be NP-complete in [2]. First, we state the path-forest problem.

Problem: Burning Path-Forest
Input: A path-forest G and an integer k.
Question: Does there exist a burning sequence $(x_1, x_2, \dots, x_k)$ for G?

Now, we formulate the cycle-forest problem.

Problem: Burning Cycle-Forest
Input: A cycle-forest G' and an integer k'.
Question: Does there exist a burning sequence $(x_1, x_2, \dots, x_{k'})$ for G'?

Theorem 5.1. *The Burning Cycle-Forest problem is NP-complete.*

Proof. Suppose that we have an instance of the path-forest problem; that is, we have a path-forest $G = (V, E)$ with components $G_1, \dots, G_m$ and a positive integer k. For each component G_i $(1 \le i \le m)$, label one of the endpoints as $G_{i,1}$, its only neighbor as $G_{i,2}$, and continue labeling unlabeled neighbors of $G_{i,j}$ as $G_{i,j+1}$, $1 \le j \le \ell_i - 1$, where $\ell_i = |G_i|$. Hence, we can write $V = \{G_{i,j} : 1 \le i \le m,\ 1 \le j \le \ell_i\}$ and $E = \{G_{i,j}G_{i,j+1} : 1 \le i \le m,\ 1 \le j \le \ell_i - 1\}$. We will construct an instance of the cycle-forest problem as follows. Define $G' = (V', E')$ with $V' = \{G'_{i,j} : G_{i,j} \in V,\ 1 \le i \le m,\ 1 \le j \le \ell_i\}$ and $E' = \{G'_{i,j}G'_{i,j+1} : G_{i,j}G_{i,j+1} \in E,\ 1 \le i \le m,\ 1 \le j \le \ell_i - 1\} \cup \{G'_{i,\ell_i}G'_{i,1} : 1 \le i \le m\}$. Then G' is a disjoint union of cycles $G'_1, \dots, G'_m$, since we add an edge between the endpoints of each G_i. Moreover, define $k' = k$. We are now ready to prove NP-completeness of the Graph Burning problem for cycle-forests.

First, assume that we are given a path-forest G with components $G_1, \dots, G_m$ and a burning sequence S of length k. Since each path G_i is a spanning tree of the corresponding cycle G'_i, Corollary 3.1 implies that $b(G'_i) \le b(G_i)$. This is true for all i, and so $b(G') \le b(G)$. It follows that S is a burning sequence for the cycle-forest G'.

Conversely, assume that G' has a burning sequence S' of length k'. Suppose each G'_i burns in k'_i rounds. It suffices to show that from each G'_i, we can remove an edge to obtain a path G_i which burns in $k_i = k'_i$ rounds. Regarding vertices which are burned in round k'_i, there are two possible cases: either such a vertex has zero neighbors which are burned in the k'_ith round, or it has at least one.

Case 1: There is a vertex in G'_i which is burned in round k'_i, such that its neighbors are not. Without loss of generality, suppose this is the vertex G'_{i,ℓ_i}. If G'_{i,ℓ_i} is a source, then by definition its burning round does not depend on its neighbors. Hence, deleting the edge $G'_{i,\ell_i}G'_{i,1}$ creates the path G_i which can be burned in k_i rounds, using the same burning sequence as G'_i. If G'_{i,ℓ_i} is not a source, then both of its neighbors are burned in round $k'_i - 1$. Indeed, if one neighbor were burned in round $k'_i - j$ for some $j \ge 2$, then G'_{i,ℓ_i} would be burned in round $k'_i - j + 1 < k'_i$ which is a contradiction. But now we may once again delete the edge $G'_{i,\ell_i}G'_{i,1}$, as the neighbor G'_{i,ℓ_i-1} guarantees that G'_{i,ℓ_i} is burned in round k'_i. This creates the path G_i which can be burned in k_i rounds.

Case 2: There are two vertices in G_i' which are neighbors and both have burning round k_i'. Without loss of generality, suppose these vertices are G'_{i,ℓ_i} and $G'_{i,1}$. Since G'_{i,ℓ_i} and $G'_{i,1}$ are burned in the same round, they do not affect each other. That is, the edge $G'_{i,\ell_i}G'_{i,1}$ does not contribute to the burning sequence and may be removed. The newly created path G_i can be burned in k_i rounds according to the same burning sequence as G_i'.

From both cases it follows that if G_i' can be burned in k_i' rounds, it is guaranteed that the corresponding path G_i can be burned in k_i rounds.

By making the reduction from the path-forest problem to the cycle-forest problem, we have shown that a path-forest can be burned in k rounds if and only if a cycle-forest can be burned in k rounds. We conclude that the Burning Cycle-Forest problem is NP-complete. □

6

Conclusion & Discussion

In this thesis, we studied graph burning using two approaches: we made estimations for the burning number of necklace graphs and studied the complexity of finding upper-bound paths and burning cycle-forests.

In Chapter 4, we provided a lower bound and upper bound for the burning number of necklace graphs. Necklace graphs were constructed to have a path-like structure by concatenating pearls. In order to relate the burning number of necklaces to known burning numbers, lower-bound paths and upper-bound paths were introduced. It was shown that such paths for necklaces can be constructed by connecting lower-bound paths or upper-bound paths of pearls in series. Proofs were given that lower-bound paths provide a lower bound for the burning number of pearls, and upper-bound paths an upper bound. These results led to the main result: a lower bound and upper bound for the burning number of necklaces. Furthermore, necklaces whose pearls are perfect have a burning number which can take only two values.

One other important result from Chapter 4 is the NP-completeness of finding upper-bound paths, which was proven by making a reduction from the Hamiltonian Path problem. In Chapter 5, an NP-completeness proof was given for the graph burning problem for cycle-forests by reducing from the graph burning problem for path-forests.

A future line of research could focus on characterizing graphs that contain upper-bound paths; the results derived in this thesis might then contribute to verifying the Burning Number Conjecture for this class of graphs. In addition, the development of an approximation algorithm to find an upper-bound path is a problem that could be studied. Finally, the concept of constructing path-like graphs and estimating the burning number using paths could be applied to different classes of graphs. For instance, a necklace of k-regular pearls might be a useful construction to provide an upper bound for the burning number of all k-regular graphs, as its path-like structure leads to a high burning number.

Bibliography

[1] Paul Bastide et al. “Improved Pyrotechnics: Closer to the Burning Number Conjecture”. In: *The Electronic Journal of Combinatorics* 30.4 (2023). DOI: 10.37236/11113.

[2] Stéphane Bessy et al. “Burning a graph is hard”. In: *Discrete Applied Mathematics* 232 (2017), pp. 73–87. DOI: 10.1002/andp.19063240204.

[3] Anthony Bonato, Jeannette Janssen, and Elham Roshanbin. “How to Burn a Graph”. In: *Internet Mathematics* 12 (2015), pp. 85–100. DOI: 10.1080/15427951.2015.1103339.

[4] Anthony Bonato and Shahin Kamali. *Approximation Algorithms for Graph Burning*. Lecture Notes in Computer Science. Springer, Cham, 2019, pp. 74–92. DOI: 10.1007/978-3-030-14812-6_6.

[5] Anthony Bonato and Thomas Lidbetter. “Bounds on the burning numbers of spiders and path-forests”. In: *Theoretical Computer Science* 794 (2019), pp. 12–19. DOI: 10.1016/j.tcs.2018.05.035.

[6] Felipe de Carvalho Pereira et al. “Solving the Graph Burning Problem for Large Graphs”. 2024. DOI: 10.48550/arXiv.2404.17080.

[7] Reinhard Diestel. *Graph Theory*. 6th ed. Springer Berlin, Heidelberg, 2025. DOI: 10.1007/978-3-662-70107-2.

[8] Zahra Rezai Farokh et al. “New heuristics for burning graphs”. 2020. DOI: 10.48550/arXiv.2003.09314.

[9] Shannon Fitzpatrick and Leif Wilm. “Burning Circulant Graphs”. 2018. DOI: 10.48550/arXiv.1706.03106.

[10] Jesús García-Díaz, José Alejandro Cornejo-Acosta, and Joel Antonio Trejo-Sánchez. “A greedy heuristic for graph burning”. In: *Computing* 107.91 (2025). DOI: 10.1007/s00607-025-01436-9.

[11] Michael Garey and David Johnson. *Computers and Intractability: A Guide to the Theory of NP-completeness*. W.H. Freeman and Company, 1979, p. 60. DOI: 10.1007/978-3-662-70107-2.

[12] Arya Tanmay Gupta, Swapnil Lokhande, and Kaushik Mondal. *Burning Grids and Intervals*. Lecture Notes in Computer Science. Springer, Cham, 2021, pp. 66–79. DOI: 10.1007/978-3-030-67899-9_6.

[13] Michaela Hiller, Eberhard Triesch, and Arie Koster. *On the Burning Number of p-Caterpillars*. AIRO Springer Series. Springer, Cham, 2020, pp. 145–156. DOI: 10.1007/978-3-030-63072-0_12.

[14] Shahin Kamali, Avery Miller, and Kenny Zhang. *Burning Two Worlds*. Lecture Notes in Computer Science. Springer, Cham, 2020. DOI: 10.1007/978-3-030-38919-2_10.

[15] Yinkui Li, Guiyu Shi, and Xiaoxiao Qin. “Graphs with burning number three”. In: *Applied Mathematics and Computation* 487 (2018). DOI: 10.1016/j.amc.2024.129100.

[16] Huiqing Liu, Ruiting Zhang, and Xiaolan Hu. “Burning number of theta graphs”. In: *Applied Mathematics and Computation* 361 (2019), pp. 246–257. DOI: 10.1016/j.amc.2019.05.031.

[17] Dieter Mitsche, Paweł Prałat, and Elham Roshanbin. “Burning number of graph products”. In: *Theoretical Computer Science* 746 (2018), pp. 124–135. DOI: 10.1016/j.tcs.2018.06.036.

[18] Yukihiro Murakami. “The Burning Number Conjecture is True for Trees without Degree-2 Vertices”. In: *Graphs and Combinatorics* 40.82 (2024). DOI: 10.1007/s00373-024-02812-6.

[19] Kai An Sim, Ta Sheng Tan, and Kok Bin Wong. “On the Burning Number of Generalized Petersen Graphs”. In: *Bulletin of the Malaysian Mathematical Sciences Society* 41 (2018), pp. 1657–1670. DOI: 10.1007/s40840-017-0585-6.