PACIFIC INSTITUTE FOR THE MATHEMATICAL SCIENCES VIRTUAL EXPERIMENTAL MATHEMATICS LAB (PIMS VXML) FINAL REPORT: EXPLORING BOXICITY: GRAPH REPRESENTATIONS IN MULTIDIMENSIONAL SPACES

AMNA ADNAN*[1], MATTHEW BARCLAY*[2], MARCO CAODURO†[3], JOSHUA CHILDS*[2], WILL EVANS†[3], AND TAO GAEDE‡[2]

1. Introduction

1.1. Introduction to Boxicity.

An axis-parallel box in $\mathbb{R}^d$ is a Cartesian product $I_1 \times I_2 \times \cdots \times I_d$, where, for $i \in \{1, 2, \ldots, d\}$, I_i is an interval of the real line. For a graph $G = (V, E)$, the *boxicity* of G, denoted by box(G), is the minimum dimension d such that G is the intersection graph of a family $(B_v)_{v \in V}$ of d-dimensional boxes.

Definition 1.1.1 (Intersection Graph). *Let $\mathcal{I} = \{I_1, \ldots, I_n\}$ be a family of sets. The* intersection graph *of $\mathcal{I}$ has a vertex v_j for each $I_j \in \mathcal{I}$, and two vertices v_j, v_k are adjacent if and only if $I_j \cap I_k \neq \emptyset$.*

Boxicity was first introduced by Fred S. Roberts in 1969 [17] alongside cubicity, a similar parameter where the (hyper-)boxes are replaced by d-dimensional (hyper-)cubes with unit side length. It has been extensively studied since then, see for example [1, 6, 7, 10, 11, 20].

1.2. Motivation.

Boxicity is a continuation and alternative approach to J.E. Cohen's ecological niche competition graphs [16]. It arose out of their shared interest in applications of combinatorics in biology and the social sciences [18]. Later, in 1981, Roberts and R.J. Opsut [15] found additional uses in the geometric logistics of airplane fleet maintenance.

Boxicity acts as a measure of both the representational complexity of a system and the dimension of interdependence within a system, such

Date: November 2023 – May 2024.
*Undergraduate Authors. †Faculty Mentor. ‡Graduate Mentor.
[1]University of Calgary, Calgary, Alberta.
[2]University of Victoria, Victoria, BC.
[3]The University of British Columbia, Vancouver, BC.

as biological niches, predator-prey relationships, task assignments, and maintenance schedules [15, 16]. The higher the boxicity, the more dimensions are needed to represent the system unambiguously. This can be useful in organizing work that must be done in real space. If a system is instantiated in a 2-dimensional floor plan but the system's boxicity is at least 4 there are more likely to be interferences within the system. This is because components of the system will be overlapping that are not necessarily interdependent.

In addition to its real-world applications, boxicity is interesting from a purely theoretical point of view as a graph invariant with direct ties to Euclidean geometry.

1.3. **The Initial Problems.** During our PIMS VXML project, our supervisor provided us with three initial problems to guide our research progress. Our supervisor also directed a literature review and provided exercises to improve our understanding of the concepts and tools used in the study of boxicity. We then subsequently learned different methodologies for finding the boxicity and bounds for the boxicity of a graph.

1. The Boxicity of the Kneser Graph $\mathrm{K}(7,3)$. A Kneser graph $\mathrm{K}(n,k)$ is a graph whose vertices correspond to the k-element subsets of a set of n elements, and where two vertices are adjacent if and only if the two corresponding sets are disjoint.

Motivation for why the boxicity of $\mathrm{K}(7,3)$ can be traced back to Caoduro's PhD thesis [2]: The Peterson graph $\mathrm{K}(5,2)$ is a familiar graph for most, because it often serves as a useful example and counterexample for many problems in graph theory. In the thesis, it is proved that the boxicity of the Peterson graph is three, and that the approach can be extended to the line graphs (*vis.* Definition 2.1.7) of complete graphs, $L(K_n)$, for $n \geq 2$. This prompted further study of $\mathrm{box}(L(K_n))$, which corresponds to $\mathrm{K}(n,2)$. It was proved by Caoduro and Sebő [4] that Kneser graphs of the form $\mathrm{K}(n,2)$ have boxicity $n-2$ for all $n \geq 5$. To push for a more general result, of $box(\mathrm{K}(n,k))$, it would be fruitful to focus on the next smallest nontrivial graph: $\mathrm{K}(7,3)$.

2. The Boxicity of the Mycielskian of a Cycle. Another motivating problem was proving that the boxicity of the Mycielskian (*vis.* Definition 5.0.1) of a cycle, $\mathrm{M}(C_k)$, was equal to 3 for all $k \geq 5$.

Motivation for why the boxicity of the Mycielskian is interesting is outlined by Kamibeppu in [12]: An implication of a theorem relating the boxicity of a graph and its chromatic number, proved by Chandran et al. [5], was that if $\mathrm{box}(G)$ is close to the maximum boxicity, then the chromatic number of the graph must be large. As Kamibeppu [12] explains, it is a natural question to wonder about the boxicity of the Mycielskian of a graph, because the chromatic number of the Mycielskian of G is more than that of G.

3. The Boxicity of the Complement of a Tree. Finally, we explored the boxicity of complements of trees and similar structures. Given that computing the boxicity of a graph is NP-hard, as shown by Cozzens [8], our goal is to find non-trivial graph classes for which it is possible to compute boxicity in polynomial time.

To work on these problems and other research interests, we learned different methodologies for finding the boxicity of a graph.

1.4. **Overview.** In Section 2, we define fundamental graphs, graph structures, graph operations, and the tools used in our study of boxicity. We also explain a construction taken from Caoduro and Sebő [4] and how it is used to find the upper bound of the boxicity of a graph. There are four theoretical results and explorations covered, while the bulk of the computational progress is focused on Kneser graphs. Section 3 covers an algorithm used to compute the boxicity of the complement of a tree. This relies on a construction called *ants* being maximal interval-order subgraphs for trees. Section 4 offers a similar result using *crabs*, another construction we define, coverings to find the boxicity of the complement of line graphs of trees. Section 5 covers the exploration into the boxicity of the Mycielskian (*vis.* Definition 5.0.1) of a cycle, $\mathrm{M}(C_k)$, and Section 6 an investigation inspired by the exploration of $\mathrm{M}(C_k)$. The computational work on the boxicity of the Kneser graph is covered in Section 7, and the report closes by discussing future directions in Section 8.

2. Background and Methods

We discuss the tools used in our study of boxicity, and review important classes of graphs that we explored in our research. All graphs herein are assumed to be simple and undirected, that is, there is at most one edge between any two vertices, and no edges are loops connected to only one vertex.

2.1. **Basic Graph Structures.** Some fundamental definitions used in graph theory. Let $G = (V, E)$ be a graph.

Definition 2.1.1 (Complete Graph). *G is a* complete graph *if $uv \in E$ for each pair of distinct vertices $u, v \in V$. A complete graph on n vertices is denoted by K_n.*

Definition 2.1.2 (Induced Subgraph). *Let $U \subset V(G)$. The* induced subgraph of U on G *is the graph H on the vertex set U, where for each edge $uv \in E(G)$, $uv \in E(H)$ if and only if $u, v \in U$.*

Definition 2.1.3 (Clique). *A subset $Q \subset V(G)$, is a* Clique *if $Q = K_n$ as an induced subgraph of G for some n.*

Definition 2.1.4 (Independent Set). *Let $S \subset V(G)$. S is an* independent set *if for every pair of vertices $u, v \in S$, $uv \notin E$.*

Definition 2.1.5 (Neighbourhood). *Let $v \in V(G)$. The* open neighbourhood *of v, denoted $N(v)$, is the set of all vertices adjacent to v. The* closed neighbourhood *of v, denoted by $N[v]$, is the set $N(v) \cup \{v\}$.*

We can also modify common graph classes in several ways to produce new graphs; we investigate the boxicity of several such derived classes in future sections of this report.

Definition 2.1.6 (Complement). *The* complement *of a graph $G = (V, E)$ is a graph $\overline{G} = (V, \overline{E})$. The complement graph has the same vertex set as G, but a complementary edge set. The edge set is defined as*

$$E(\overline{G}) := \{uv : u, v \in V(G) \text{ and } uv \notin E(G)\}.$$

As highlighted in Subsection 2.3, when studying the boxicity of a graph G we are almost always working in the complement graph $\overline{G}$.

Definition 2.1.7 (Line Graph). *Let $G = (V, E)$ be a graph. The* line graph *of G, denoted $L(G)$, is the graph with vertex set*

$$V(L(G)) = \{v_e \mid e \in E(G)\},$$

and for two vertices $v_e, v_f \in V(L(G))$, $v_e v_f \in E(L(G))$ if and only if e and f are adjacent (share a vertex) in G.

We study line graphs of trees in Subsection 3.

2.2. **Interval-Order Subgraphs.**

Definition 2.2.1 (Interval Graph). *Let $G = (V, E)$ be a graph. G is an* interval graph *if G is the intersection graph of line segments in the real line. For all $u \in V(G)$ assign an interval $I_u \subseteq \mathbb{R}$ such that $I_u \cap I_v \neq \emptyset$ if and only if $uv \in E(G)$.*

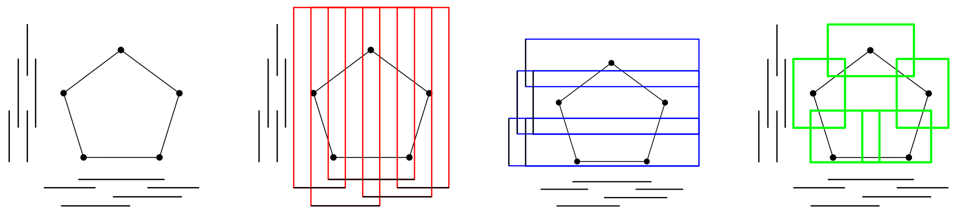


FIGURE 1. Example of an interval box construction on C_5

Interval graphs are one-dimensional box representations of graphs; if a graph has boxicity 1, it is an interval graph. In this way boxicity is the k-dimensional extension of interval graphs. A box-intersection representation of a graph G may be constructed by the intersection of interval graphs. This is described in the following theorem.

Theorem 2.1. *A graph G has boxicity at most k if and only if it can be represented as the intersection graph of k interval graph representations.*

See Figure 1.

Depending on the graph you are working with, taking these intersections can be difficult. Intersections are reductive, not constructive, requiring a reasonable estimate of a graph's boxicity. Computationally working with intersections often necessitates nested recursive searches, which are undesirable as they may greatly increase an algorithm's run time.

To avoid these challenges we want a tool for finding boxicity that works on a graph directly, and is constructive.

Definition 2.2.2 (Co-Interval Graph)**.** *A graph G is* co-interval *if it is the complement of an interval graph.*

Definition 2.2.3 (Interval-Order Subgraph [4])**.** *Let $G = (V, E)$ be a graph, and let $\sigma = (v_1, v_2, \ldots, v_n)$ be an ordered permutation of $V(G)$. We construct an* interval-order *subgraph of G, $G^\sigma = (V, E^\sigma)$, as follows: let $V_0 = V(G)$, and let $V_i = V_{i-1} \cap N_G(v_i)$ for $i \in \{1, \ldots, n\}$. We can think of V_i as the set of live vertices at step i; we thus call V_i the* live set*. The edge set is $E^\sigma = E_1 \cup E_2 \cup \ldots \cup E_{n-1} \subseteq E(G)$, where each E_i is the set of edges $\{v_i u \mid u \in V_i\}$; that is, at step i we add to E^σ the edges between v_i and each vertex in the live set V_i.*

For any ordering σ of $V(G)$, G^σ is a co-interval subgraph of G [4]. For this reason, co-interval subgraphs will be hereafter referred to as interval-order subgraphs. See Figure 2.

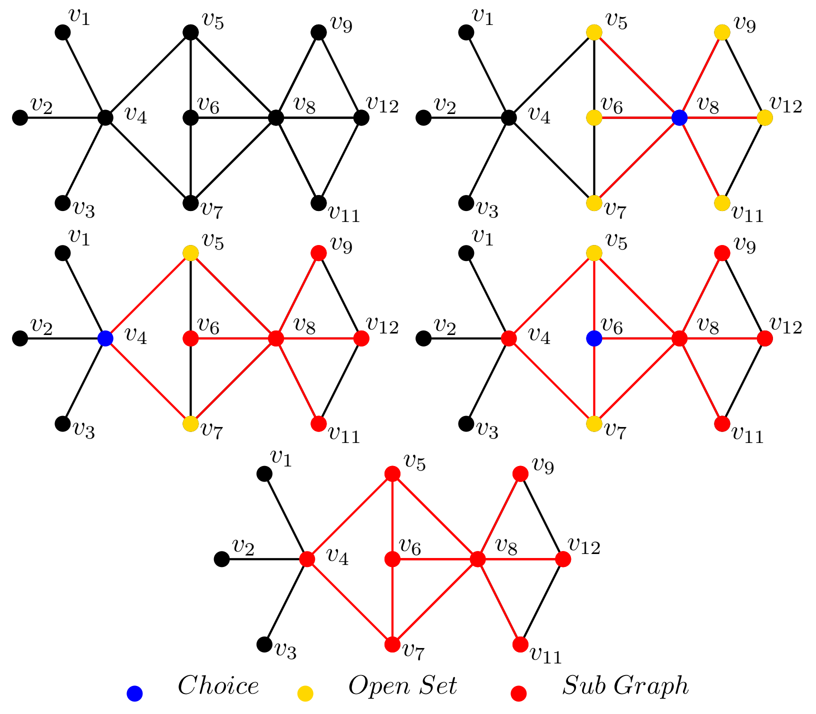


FIGURE 2. Example of an interval order construction

When utilizing the above construction, there are often vertices near the end of an ordering that are 'left out', so to speak, by not being connected to the body of the subgraph. Thankfully, these isolated vertices do not affect the process.

Lemma 2.2 (Cozzens and Roberts, 1983 [9]). *If G is an interval-order graph, then G plus k isolated vertices is an interval-order graph.*

Proof. Let H be the graph containing G and k isolated vertices. Since $\overline{G}$ is an interval graph, $\overline{H}$ must be as well; each isolated vertex in H is adjacent to all other vertices in $\overline{H}$, so it is trivial to extend the interval representation of $\overline{G}$ to an interval representation of $\overline{H}$. □

This means that we can assume any interval-order subgraph of G contains all of the vertices of $V(G)$, reducing our problem to finding spanning subgraphs whose unions cover the edges of G.

2.3. **Graph Coverings.**

We begin with some important definitions, and introduce our main tool for studying boxicity.

Definition 2.3.1 (Graph Covering). *Let $G = (V, E)$ be a graph, and let $\mathcal{F} = \{H_1, H_2, \ldots, H_k\}$ be a family of subgraphs of G. We say that*

$\mathcal{F}$ covers G *if the graph union*

$$H = H_1 \cup H_2 \cup \cdots \cup H_k,$$

defined by $V(H) = \cup_{i=1}^{k} V(H_i)$ *and* $E(H) = \cup_{i=1}^{k} E(H_i)$, *is equal to* G. *That is,* $V(H) = V(G)$ *and* $E(H) = E(G)$

Definition 2.3.2 (Interval-Order Cover). *Let* $G = (V, E)$ *be a graph, and let* $\mathcal{F} = \{H_1, H_2, \ldots, H_k\}$ *be a family of* interval-order *subgraphs of* G. *We say that* $\mathcal{F}$ *is a* k-interval-order cover *of* G *if and only if* $\mathcal{F}$ *covers* G.

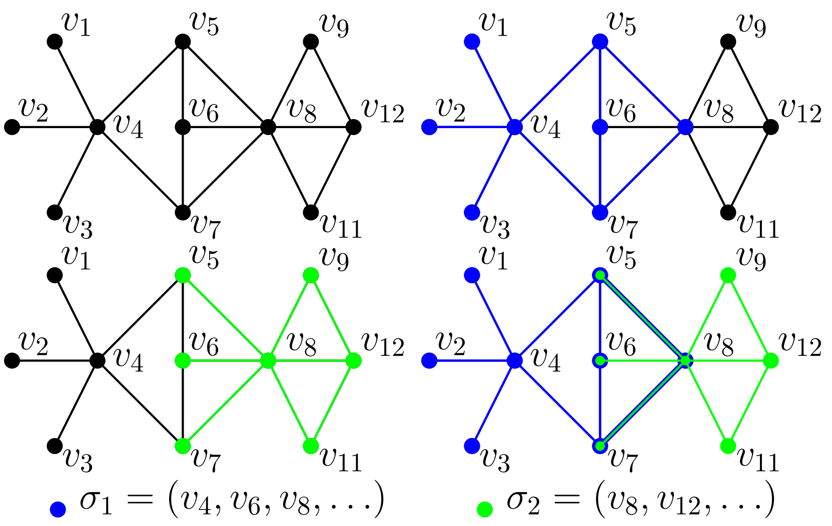

v_1
v_2
v_3
v_4
v_5
v_6
v_7
v_8
v_9
v_{11}
v_{12}
$\sigma_1 = (v_4, v_6, v_8, \ldots)$
$\sigma_2 = (v_8, v_{12}, \ldots)$

Figure 3. Example of an interval order cover

The following lemma is the cornerstone of our research. We will use this extensively to find upper bounds on the boxicity of various graphs and graph classes.

Lemma 2.3 (Cozzens and Roberts, 1983 [9]). *Let* $G = (V, E)$ *be a graph, and let* $\overline{G}$ *be its complement. Then* $\text{box}(G) \le k$ *if and only if* $\overline{G}$ *has a* k*-interval-order cover.*

Thus, when we are looking for upper bounds on $\text{box}(G)$, we wish to use the least number of interval-order subgraphs in our covering of $\overline{G}$. Hence we are only interested in such subgraphs which are *maximal*, that is, those which can not be covered entirely with another interval-order subgraph. We refer to the following definition.

Definition 2.3.3 (Maximal Interval-Order Subgraphs (MIOS)). *Let* $G = (V, E)$ *be a graph, and let* σ *be an ordering of* $V(G)$. *We say* *that G^σ is* inclusion-wise maximal *if and only if there does not exist a ordering τ of $V(G)$ such that G^σ is a proper subgraph of G^τ.*

Following the example of previous literature (see for e.g. [3], [4]), we seek ways of characterizing the maximal interval-order subgraphs of select graph classes.

If we can characterize (or better, enumerate) the maximal interval-order subgraphs of a graph, then it becomes much easier to find a minimal covering of the graph's complement. Without such characterizations, we would have to try every possible interval-order subgraph to find a potentially minimal covering.

2.4. **Vertex Orderings.** As interval-order subgraphs are built using vertex orderings, the permutations of $V(G)$ can be used to characterize these subgraphs. When building interval-order subgraphs, we often refer to *partial orderings*: incomplete lists of vertices. There are several useful lemmas that aid in finishing these partial orderings. Herein, *the algorithm* refers to 2.2.3.

Definition 2.4.1 (Partial Ordering)**.** *Let $G = (V, E)$ be a graph of order n. A* partial ordering *σ_i of $V(G)$ is a list of $i < n$ vertices. We also use σ_i to refer to the* set *of vertices in the partial ordering.*

Lemma 2.4 (Caoduro and Sebő, 2023 [4])**.** *Let Δ be the maximum degree of a graph G. Then the live set of vertices V_i is nonempty for at most Δ steps of the algorithm. That is, $V_{\Delta+1} = \emptyset$.*

Proof. Let $\sigma = \{v_1, v_2, \ldots, v_\Delta, \ldots, v_n\}$ be a vertex ordering on a graph G, and let $u \in V(G)$ be a vertex of degree Δ. Suppose u is in $N(v_1), N(v_2), \ldots, N(v_\Delta)$. Then $u \in V_\Delta$. But u has degree Δ, so it cannot be in $N(v_{\Delta+1})$. Thus $V_{\Delta+1} = V_\Delta \cap N(v_{\Delta+1}) = \emptyset$. □

Lemma 2.5 (Caoduro and Sebő, 2023 [4])**.** *If $|V_i| = 2$, then there are exactly two ways to maximally finish the partial ordering.*

Proof. Let σ_i be a partial ordering, and let $V_i = \{u, v\}$, where $u, v \notin \sigma_i$. To maximize edge-inclusion, first add all vertices in $(N(u) \cap N(v)) \setminus \sigma_i$. Then we can either add all the remaining vertices neighbouring u, or those neighbouring v. Afterwards, the live set is empty and any unused vertices can be added to the partial ordering with arbitrary order. □

3. Algorithm for Producing a Family of Ants Used to Compute the Boxicity of the Complement of a Tree T

One of our main results was a polynomial-time algorithm that produces a family of ants that can be used to compute the boxicity of the complement of a tree T. By this, we mean that by finding a minimum ant-covering of trees in polynomial time, we were able to obtain an algorithm to compute the boxicity of the complement of a tree.

Note: Proofs will be presented in an extended version of this report under preparation.

3.1. **Preliminaries.** We define a tree graph in the standard way:

Definition 3.1.1 (Trees and Forests). *A graph $T = (V, E)$ is called a* tree *if it is connected and acyclic. A* forest *is an acyclic graph, one where every disconnected component is a tree.*

Hereafter, a graph labelled T will always refer to a tree.

Definition 3.1.2 (Leaf). *For a tree T, a vertex of degree 1 is a leaf.*

3.2. **Characterizing the Maximal Interval-Order Subgraphs.** From Lemma 2.3, we know that $\text{box}(G) \leq k$ if and only if $\overline{G}$ has a k-interval-order cover, Definition 2.3.2. So, to determine the upper bound on $\text{box}(G)$, we need to cover the edges of $\overline{G}$ with as few maximal interval-order subgraphs as possible.

The maximal interval-order subgraphs for trees are completely characterized by a construction we define as *ants*.

Definition 3.2.1 (uv-ant). *Let $G = (V, E)$ be a graph and $uv \in E$. A uv-ant with respect to G is the graph A having vertex-set $V(A) := N(u) \cup N(v)$ and edge set $E(A) := \delta(u) \cup \delta(v)$. We call the edge uv the* body *of the ant and the edges $E(A) \setminus \{uv\}$ the* legs *of the ant. See Figure 4 for an illustration.*

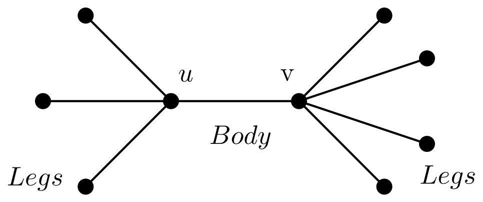


FIGURE 4. Example of a uv-ant with the legs and body labelled

We use the following Lemma 3.1 from Cozzens and Roberts [9]:

Lemma 3.1 (Cozzens and Roberts [9]). *Let G be a graph, and let $uv \in E(G)$. Then the uv-ant is an interval-order subgraph.*

Essentially, we can completely characterize the maximal interval-order subgraphs of trees, they turn out to be ants.

Lemma 3.2. *Let H be a subgraph of T. If H is a maximal interval-order subgraph of T, then H is a uv-ant for some $uv \in E(T)$.*

Now that we can characterize the maximal interval-order subgraphs for trees as ants, Definition 3.2.1, it becomes much easier to find a minimal covering with maximal interval-order subgraphs, ants 3.2.1, of $\overline{T}$.

3.3. **Finding a Minimal Covering with Maximal Interval-Order Subgraphs, Ants 3.2.1, of $\overline{T}$.**

We define the ant-packing number and the ant-covering number of a graph G by the following:

Definition 3.3.1 (Ant-Packing Number)**.** *Let G be a graph. The ant-packing number of G, denoted by $\alpha_A(G)$, is the maximum number of pairwise edge-disjoint ants in G.*

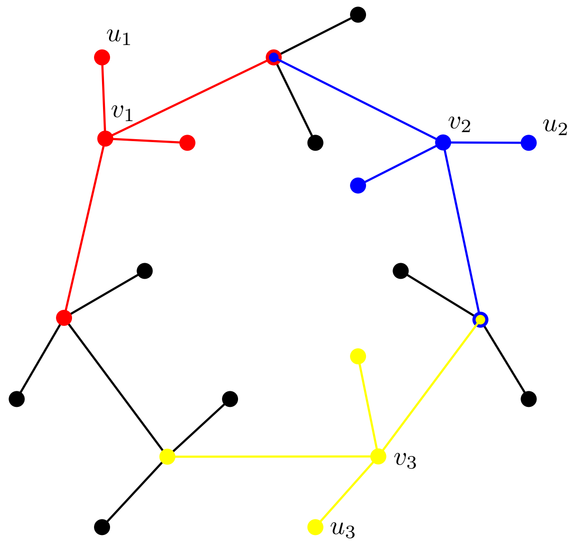


Figure 5. Example of an ant-packing of a graph G

Definition 3.3.2 (Ant-Covering Number)**.** *Let G be a graph. The ant-covering number of G, denoted by $\tau_A(G)$, is the minimum number of ants needed to cover the edges of G.*

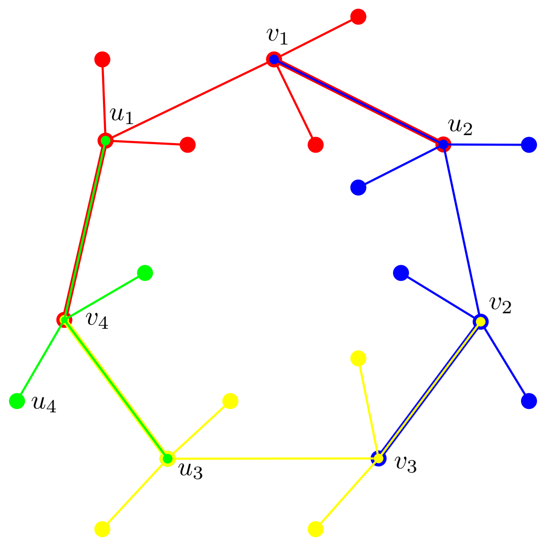


FIGURE 6. Example of an ant-covering of a graph G

As mentioned before, generally for a graph, G, the covering, ν_A, is at least the packing, τ_A, i.e. $\tau_A \leq \nu_A$. We can see why the ant-covering number, $\tau_A(G)$, is upper-bounded by the ant-packing number, $\alpha_A(G)$, using Figure 5 and Figure 6. Since each ant in a packing must be disjoint, the number of ants in a maximal packing $\alpha_A(G)$ is bounded above by the number of ants in a minimal covering τ_A, where overlaps are allowed. The covering, ν_A, and packing, τ_A, are not necessarily equal, but in trees they are.

The relationships between the boxicity of the complement of T, box($\overline{T}$), and the ant-packing number, $\alpha_A(G)$, and the ant-covering number, $\tau_A(G)$, are outlined by the following Lemma:

Lemma 3.3. *Let T be a tree. Then,* $\nu_A(T) \leq \text{box}(\overline{T}) \leq \tau_A(T)$.

We will see that the aforementioned algorithm proves that for a tree, the ant-packing number and the ant-covering number are equal:

Theorem 3.4. *Let T be a tree, then $\alpha_A(T) = \tau_A(T)$.*

The proof of Theorem 3.4 is algorithmic. Since for a tree T, $box(\overline{T}) = \alpha_A(T) = \tau_A(T)$, this algorithm can be used to compute the boxicity of the complement of a tree in polynomial time.

Now, *because for a tree* T, $\alpha_A(T) \leq box(\overline{T}) \leq \tau_A(T)$ and the algorithm proves that $\alpha_A(T) = \tau_A(T)$, we have $box(\overline{T}) = \alpha_A(T) = \tau_A(T)$.

Therefore as a Corollary of Theorem 3.4, we can compute the boxicity of the complement of a tree in polynomial-time.

Theorem 3.5 (Corollary of Theorem 3.4)**.** *Using Algorithm 3.4, we can compute the boxicity of the complement of* T, $box(\overline{T})$, *in polynomial time.*

3.4. **Algorithm.** The algorithm makes mention of "almost-leaves", which can be defined as follows:

Definition 3.4.1. *An almost-leaf can be defined as a vertex that becomes a leaf after all leaves of the original tree* T *are removed.*

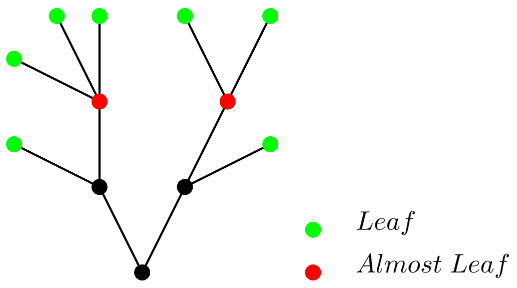


Figure 7. Example a graph G with leaves and almost-leaves

For a tree T rooted at a vertex r, we define the depth, d, of a vertex, and the height h of the tree as follows:

Definition 3.4.2. *Let* T *be a tree rooted at* r. *The depth,* d, *of a vertex,* v, *is the distance between* v *and* r.

Definition 3.4.3. *Let* T *be a tree rooted at* r. *The height,* h, *of* T *is the maximum distance between* r *and a leaf,* l, *i.e.* $h = max\{d(r,l)\}$.

Algorithm:

If the graph G is a forest, then this algorithm is performed for each connected component in parallel.

(1) Choose an arbitrary vertex, r, and root the tree, T, at r.

For a connected component, if the height, h, is at most 2, then for each depth 1 vertex, there will be an ant for the covering and packing, therefore we can see that $\alpha_A(T) = \tau_A(T)$.

(2) For the tree T, rooted at r, select all almost leaves furthest away from the root vertex.
(3) Label each almost leaf, $u_{h,i}$, where h is the height of the tree, and i is the almost-leaf count for the tree T.
(4) Label the almost leaf, $u_{h,i}$, and choose a non-leaf neighbour of $u_{h,i}$ and label it $v_{h,i}$. Subsequently, choose a neighbour of $u_{h,i}$ which is a leaf and call it $l_{h,i}$.

In summary, $u_{h,i}$ is an almost-leaf, $v_{h,i} \in N[u_i]$ is not a leaf, and $l_{h,i} \in N[u_i]$ is a leaf.

(5) Use u and v to construct an u, v-ant.
(6) We define a class C, $C := C \cup \{u_i, v_i\}$, where the bodies of the ants used for *covering* are stored; and a class P, $P := P \cup \{u_i, l_i\}$, where the bodies of the ants used for *packing* are stored.

For each almost-leaf, we get one packing ant and one covering ant.

(7) Now define T_{h-1} as T_h minus the edges of the $u_{h,i} - v_{h,i}$ ant.
(8) Repeat this process.

We prove that the algorithm produces a minimum ant-covering and a maximum ant-packing in an extended version of the report that is under preparation.

Because the ant-packing and ant-covering are equal for trees, we have a minimal ant-cover.

Recall that ants are the maximal interval-order subgraphs for trees. Given that we have obtained a minimal maximal interval-order subgraph covering, we know the boxicity of the complement of the tree, T, *because for a tree* T, $\alpha_A(T) \leq box(\overline{T}) \leq \tau_A(T)$ and the algorithm proves that $\alpha_A(T) = \tau_A(T)$, so we have $box(\overline{T}) = \alpha_A(T) = \tau_A(T)$. Therefore as a Corollary of Theorem 3.4, the Algorithm allows us to compute the boxicity of the complement of a tree, $box(\overline{T})$, in polynomial-time.

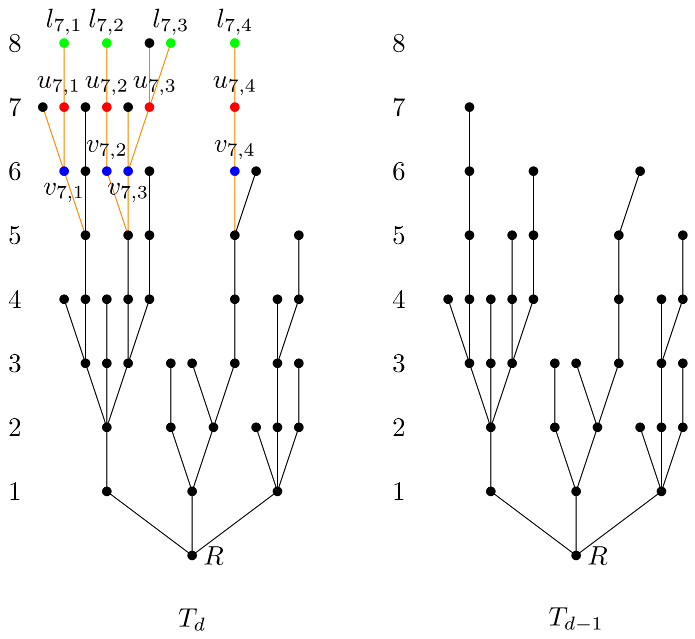


FIGURE 8. The original graph T_d with the almost leaves, $u_{d,i}$, in red, the neighbours of $u_{d,i}$ that are not leaves, $v_{d,i}$, in blue, and the neighbours of $u_{d,i}$ that are leaves, $l_{d,i}$, in green. The uv-ants constructed from the u's and v's are in orange (these are maximal interval-order subgraphs for a tree). T_{d-1} is the original graph minus the constructed ants.

4. Line Graphs of Trees

We have a similar result to the ant coverings of trees using crab coverings to find the boxicity of the complement of line graphs of trees. When working on the ant-packing-covering algorithm a nice proof of the existence of such an algorithm was outlined in parallel by Tao Gaede. This proof iterated over the line graph, and was useful in showing the following result.

4.1. **Line Graphs of Trees.** After looking at one class of graph it may be natural to consider their line graphs, as all graphs on greater than four vertices have a unique bijection to their line graph. The line graph of a tree appears similar to a tree; instead of branching edges, it consists of branching cliques. This visual similarity was believed to have significance with respect to boxicity as cliques are often included in a MIOS.

Definition 4.1.1 ((u,v,K)-Crab)**.** *Let $G = (V, E)$ be a non-complete graph, let K be a clique in G, and $u, v \in V(K)$. A* (u,v,K)-crab *with respect to G is the graph C having vertex-set $V(C) = N[u] \cup N[v]$ and the edge set $E(C) = \delta(u) \cup \delta(v) \cup E(K)$. We call the clique K the* body *of the crab and the edges $E(C) \setminus E(K)$ the* legs *of the crab.*

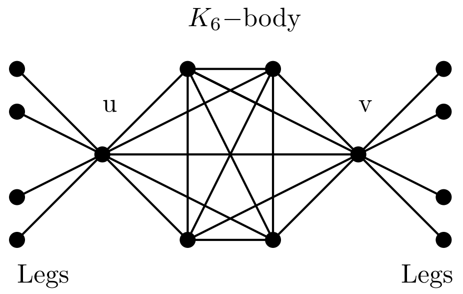


Figure 9. Example of a Crab

Lemma 4.1. *Let G be a subgraph of $L(T)$. If G is a maximal interval-order subgraph of $L(T)$, then G is a K-crab for $K \subset L(T)$ and K a clique.*

Theorem 4.2 (Existence of Polynomial Time Algorithm to determine the Boxicity of the Co-Line graph of a Tree)**.** *Given a tree T, there exists a polynomial time algorithm which determines the boxicity of $\overline{L(T)}$, the complement of its line graph.*

A detailed proof of this will appear in an extended upcoming manuscript. This follows a similar algorithm to that of the ant covering and packing shown in Subsection 3.4.

5. The Mycielskian

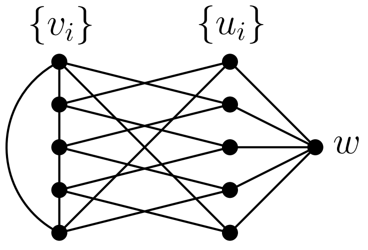
Definition 5.0.1 (Mycielskian [14])**.** *Let $G = (V, E)$ be a graph. The* Mycielskian *of G, denoted* M(G) *or $\mu(G)$, is derived as follows: let*

$V(G) = \{v_1, v_2, \ldots, v_n\}$, and create a copy $V'(G) = \{u_1, u_2, \ldots, u_n\}$. The vertices of $\mathrm{M}(G)$ *is the set*

$$V(\mathrm{M}(G)) = V(G) \cup V'(G) \cup \{w\}\,.$$

For each $u_i \in V'(G)$, $i \in \{1, \ldots, n\}$, we define $E_i(G) = \{u_i v_j \mid v_j \in N(v_i)\}$, and define $E_0 = \{wu_i \mid u_i \in V'(G)\}$. Then the edges of $\mathrm{M}(G)$ *are*

$$E(\mathrm{M}(G)) = E(G) \cup E_1(G) \cup \cdots \cup E_n(G) \cup E_0(G).$$



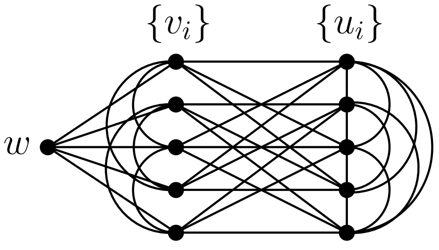


FIGURE 10. Example of the Mycielskian of C_5 and its complement

5.1. **The Mycielskian of Cycles.** The initial exploration into the problem of boxicity of the Mycielskian of a cycle, $\mathrm{M}(C_k)$, was equal to 3 for all $k \geq 5$ consisted of finding the maximal interval-order subgraphs for the complements of $\mathrm{M}(C_k)$ for $k \leq 5$. We obtained a covering of two maximal interval-order subgraphs for $\overline{\mathrm{M}(C_3)}$, and $\overline{\mathrm{M}(C_4)}$. For $\overline{\mathrm{M}(C_5)}$, there was a covering of three maximal interval-order subgraphs.

We began by attempting to prove the lower bound for $box(\mathrm{M}(C_k)) = 3$, $\forall k \geq 5$. Our two avenues of exploration consisted of:

(1) Showing that it is not possible to cover $\overline{\mathrm{M}(C_5)}$ with only 2 maximal interval-order subgraphs
(2) Bi-colouring, the method used by Caoduro in [2] to prove the lower bound for $\mathrm{M}(C_5)$: If it is supposed, for contradiction, that $box(\mathrm{M}(C_5)) \leq 2$, then by Lemma 2.3 we know $\mathrm{M}(C_5)$ can be covered by the edges of two co-interval graphs, with red edges and blue edges. The edges covered by subgraph 1 can be coloured red, and subgraph 2, blue, (not mutually exclusive). Since any induced C_5 must have boxicity 2, and we know [2] that it has no two-coloured edges, we can use the C_5 because the complement of the original C_5 subgraph is also a C_5 subgraph) in $\overline{\mathrm{M}(C_5)}$, with only an initial arbitrary colour assumption (that $\mathrm{M}(C_5)$) is two-colourable), to show that some edge in the C_5 must have two colours.

For C_k for $k \geq 6$, not all $\mathrm{M}(C_k)$ contain an induced C_5, therefore the bi-colouring method was not explored further.

5.1.1. *Mycielskian Construction Error.* When investigating the Mycielskian of a cycle a mistake in the complement where the set of v vertices $V(G) \subset V(\overline{\mathrm{M}(C_k)}$ was independent, denoted $M^*(G)$, gave rise to an edge counting argument restricting the possible maximal interval-order covers. The question became: Was it possible for a MIOS to cover more than half of the edges between $V(G)$ and $V'(G)$? If there was no MIOS 2-cover. By dividing the problem into 3 cases determined by where the order is started in the construction of a MIOS, a $v \in V(G)$ vertex, $u \in V'(G)$ vertex, or the $w \in V(M^*(G) \setminus V(G) \cup V'(G)$ vertex. This reduced the problem to 6 cases, one for each possible pair. Resulting in the following intermediate result.

Theorem 5.1. $\mathrm{box}(\mathrm{M}^*(C_k)) = 3$ *for all k.*

First it was proved that if the $V(G)$ to $V'(G)$ edges are not covered for even k then they are not for odd k. Then by contradiction on all 6 cases shown there is no possible MIOS 2-cover and a covering of all edges between $V(C_k)$ and $V'(C_k)$ for even k. For interest, the M^* construction recreated from the mistaken complement is below:

Definition 5.1.1. *For a graph $G = (V, E)$, $M^*(G)$ is the graph derived from G by following the Mycielskian construction as normal then replace $E(G)$ with $E(K_n)$, where $n = |G|$.*

The techniques derived from the M^* mistake ultimately lead to progress on finding $\mathrm{box}(\mathrm{M}(C_k))$, and new proof for $\mathrm{box}(\mathrm{M}(K_n))$

5.1.2. *Mycielskian of Cycle Progress.* In the proper construction $V(G)$ is not an independent set which makes the 6 cases to investigate more complicated as there are less restrictions on the possible MIOS. The following lemmas have been shown by deriving a contradiction using the edge counting method on 4 for the cases.

Lemma 5.2. *For $\mathrm{M}(C_k)$ there is no MIOS 2-cover of $\overline{\mathrm{M}(C_k)}$ where either order σ_1 or σ_2 has $v_1 = w$.*

Lemma 5.3. *For $\mathrm{M}(C_k)$ there is no MIOS 2-cover of $\overline{\mathrm{M}(C_k)}$ where both orders σ_1 and σ_2 have $v_1 \in V'(G)$.*

The two cases which remain are:

(1) for $v_1 \in \sigma_1$, $v_1 \in V'(G)$ and for $u_1 \in \sigma_2$, $u_1 \in V(G)$
(2) for $v_1 \in \sigma_1$, $v_1 \in V(G)$ and for $u_1 \in \sigma_2$, $u_1 \in V(G)$

The current work on the remaining two cases is being done by constructing the two orders in parallel, and tracking the edges excluded by each order as each new vertex is added then and showing that σ_1 and σ_2 cannot the partition the edge set.

5.1.3. *Mycielskian of Complete Graph Redundancy.* While the proof of the boxicity for $\mathrm{M}(C_k)$ has not been completed, work on it, the mistaken construction, and the edge counting method have led to an independent and novel proof of the boxicity of the Mycielskian of a complete graph.

Theorem 5.4. $\mathrm{box}(\mathrm{M}(K_n)) = \lfloor \frac{n}{2} \rfloor + 1$

This in-turn has led to work on split graphs as $\mathrm{M}(G) \setminus \{w\}$ is a split graph.

6. Split Graphs

Our investigation of the Mycielskian of cycles led to the Mycielskian of complete graphs and then questions about the iterated Mycielskian of complete graphs. These all involve an independent set in $\mathrm{M}(G)$ and a large clique in $\overline{\mathrm{M}(G)}$. Cliques seemed natural to focus on as they are induced subgraphs of boxicity-0. The Question of cubicity then arose, and it was quickly realized that a cube-intersection representation of a clique had a finite-content bound. This with the predictable geometry of unit-cubes, and their arrangement within a fixed volume, has led to the beginning of a classification and explicit description of the cubicity of split graphs with respect to their clique size and independent set size.

Cubicity was first introduced by Fred S. Roberts in 1969 alongside Boxicity, and is where the boxes are d-dimensional unit cubes.

Definition 6.0.1 (Unit Cube)**.** *A* unit-cube *is an axis-parallel box where all intervals I_i are of unit length.*

Definition 6.0.2 (Cubicity)**.** *Let $G = (V, E)$ be a graph and $uv \in E$. The* cubicity of G, *denoted by* $\mathrm{cub}(G)$, *is the minimum dimension d such that G is the intersection graph of a family $(C_v)_{v \in V}$ of d-dimensional unit-cubes in $\mathbb{R}^d$.*

Definition 6.0.3 (Split Graph)**.** *Let $G = (V, E)$ be a graph. G is a* split graph *if $V(G) = K \cup S$, where S is an independent set, and K is a maximal clique of G.*

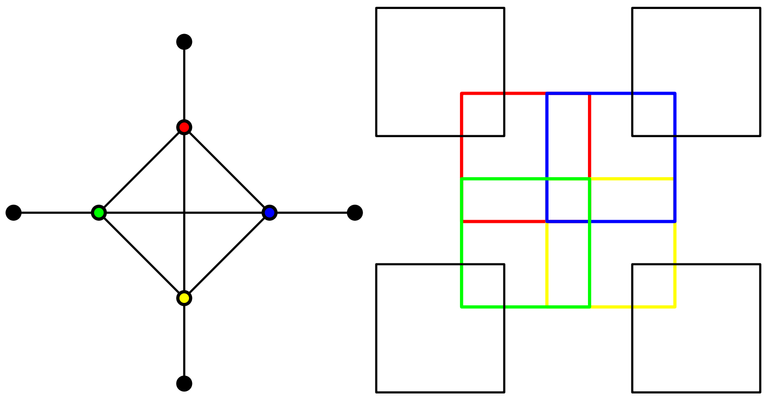


FIGURE 11. Example of a split graph and unit-cube representation

6.1. **Preliminary Clique and Independent Set Results.** Because cubicity is defined using unit cubes there is a finite volume in which to arrange the independent set vertex cubes about a clique. This realization led to the following results.

Lemma 6.1. *Let $G = K_n$ be a complete graph on n vertices. The d-dimensional representation of G is contained within a d-dimensional cube of side length 2, furthermore each k-dimensional polytope component of the cube P^k ($k = 1$ side, $k = 2$ face, $\dots$, etc.) is of k-content 2^k for $k \geq 1$.*

Lemma 6.2. *Let $G = (V, E)$ be a split graph with clique K, for $|K| = n$, independent set S, and $\operatorname{cub}(G) = d$. Then the maximum number of vertices $v \in S$ of degree $n - 1$ is $2^d - 1$.*

Lemma 6.3. *Let $G = (V, E)$ be a split graph with clique K, independent set S, and $\operatorname{cub}(G) = d$. For $|K| \geq 2^d$, the maximum number of vertices $v \in S$ is 3^d.*

6.2. **Cubicity with Cycles and Independent Sets.** The volume and dimension constraint methods as those used in complete graphs and split graphs above can be applied to cycles and cycles with independent sets the following were derived in the process of finding generalized results about graphs with cycles as induced subgraphs.

Lemma 6.4. *For C_k a cycle, given a cubicity-2 representation of C_k the maximum area contained within the cycle is*

$$\frac{1}{8}(k-4)k + 1$$

Lemma 6.5. *Let $G = (V, E)$ be a graph with* $\operatorname{cub}(G) = d$. *If $P \subset V(G)$ is a pairwise independent set of vertices then then for $Q \subset V(G)$ a pairwise independent set adjacent to all $v \in P$,*

$$|Q| + |P| \leq 2^d + 1$$

6.3. **Extending the Cubicity Results.** These preliminary results suggest that there may be a generalized method for finding the cubicity of a split graph, as well a method for finding a cubicity bound on the Mycielskian, $\mathrm{M}(G)$, for all G.

7. The Kneser Graph

7.1. **Introduction to Kneser Graphs.** One class of note are the Kneser graphs. First introduced by M. Kneser in 1955 [13], the boxicity of these graphs has been well-studied, see [2], [3], and [4].

The Kneser graph $\mathrm{K}(5, 2)$ is, in fact, isomorphic to the Petersen Graph.

Definition 7.1.1 (Kneser Graph)**.** *Let $k, n \in \mathbb{Z}^+$ such that $n \geq 2k + 1$. A* Kneser graph $\mathrm{K}(n, k)$ *is the graph with the vertex set $V = \{1, \ldots, n\}^{[k]}$, i.e. all k-subsets of $[n]$, and an edge between two vertices if and only if the respective sets are disjoint.*

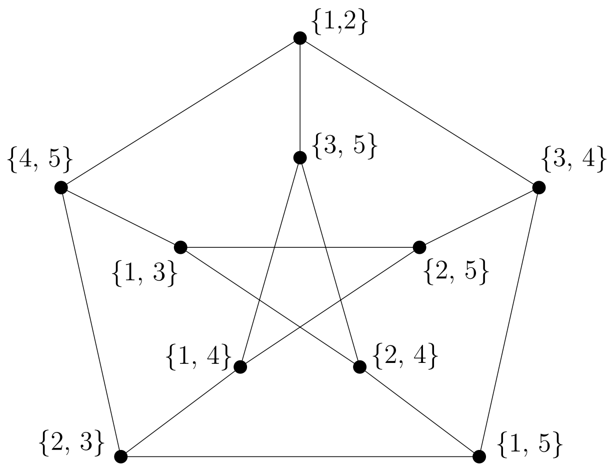


Figure 12. The Kneser Graph $\mathrm{K}(5, 2)$

Lemma 7.1 (Boxicity Upper Bound [3])**.** *Let $n \geq 2k + 1$, and let* $\mathrm{K}(n, k)$ *be a Kneser graph. Then* $\operatorname{box}(\mathrm{K}(n, k)) \leq n - 2$.

Caoduro and Sebő (2023) proved that Kneser graphs of the form $\mathrm{K}(n, 2)$ have boxicity $n - 2$ for all $n \geq 5$, improved a lower bound for large Kneser graphs with $n \geq 2k^3 - 2k^2 + 1$, and derived general upper bounds on the boxicity [4]. Furthermore, they conjectured that $\text{box}(\text{K}(n, k)) = n - k$. Hoping to further this research, we focused on the next smallest graph $\text{K}(7, 3)$.

From the literature, we know that $\text{box}(\text{K}(7, 3)) \leq 5$. Proving the upper bound is easy; one merely needs to find a k-interval-order covering of the complement. However, the lower bound is much more intractable. To prove that $\text{box}(\text{K}(7, 3)) > 4$, one must show that any combination of four maximal interval-order subgraphs cannot cover all of the edges of the graph; that is, five is required.

Hereafter, define $G := \overline{\text{K}(7, 3)}$, the complement graph of $\text{K}(7, 3)$. Subgraphs of G will be referred to by their vertex orderings.

The primary goal of this work was to categorize the maximal interval-order subgraphs of G. The main difficulty with this endeavour was the sheer number of possible vertex orderings that can produce subgraphs; as G is 30-regular, there are $\binom{35}{30}$ possible orderings. Thus it was imperative to find ways of discarding obviously non-inclusion-wise-maximal orderings.

In addition, as the Kneser graphs are vertex-transitive and symmetric, we can group together completed interval-order subgraphs by their isomorphism classes, further reducing the number of subgraphs we need to categorize and check for maximality.

We approached the problem from several different angles, trying algorithmic, exhaustive, and novel search methods. We discuss one such method here.

7.2. **The Greedy Algorithm.** Our first approach to finding maximal interval-order subgraphs was a simple edge-greedy method, where at each step of the construction we choose the vertex that would add the greatest number of new edges to the incomplete subgraph. When building the orderings, we used the following algorithm.

Let $k < 30$ and let $\sigma_k = (v_1, \dots, v_k)$ be a partial ordering. Define $R_k = V(G) \setminus \{\sigma_k\}$ to be the remaining (unused) vertices. We select v_{k+1} as follows:

1. For each unused vertex u_j in R, define $d_j = |V_k \cap N(u_j)|$ to be the *out-degree* of u_j.
2. Let $D = \{d_j \ : \ u_j \in R\}$ be the set of all out-degrees in R. Let $d_{max} = \max(D)$.
3. Append the vertex u_j with $d_j = d_{max}$ to σ_k, creating σ_{k+1}. If multiple vertices have maximum out-degree, select one arbitrarily.

This produced a number of large interval-order subgraphs, no two of which were isomorphic. We believe the multitude of non-isomorphic graphs is due, at least in part, to inherent biases in the underlying data structures. The vertices were stored as ordered lists, and when the program chose an 'arbitrary' vertex, it tended to select those stored near the end of the list due to how the search was coded.

We conjecture that this algorithm always produces a maximal interval-order subgraph, but we were unable to find a succinct proof on the matter as of the time of writing. We hope to either prove this conjecture or find a counterexample in future work.

Of note is the number of non-isomorphic (maximal) interval-order subgraphs we found for $\mathrm{K}(7,3)$. Caoduro and Sebő found that there were a total of four non-isomorphic maximal interval-order subgraphs of $\mathrm{K}(5,2)$, which could be easily characterized [4]. This seems to suggest that the $k = 3$ case is much more complex than the $k = 2$ case, and novel methods will need to be developed to further progress on this problem, as the current tools used to prove the boxicity of $\mathrm{K}(n,2)$ graphs are insufficient to tackle this much larger problem.

Code can be found in Appendix A. All functions were written in SageMath 10.2, using standard libraries.

8. Future directions

It was shown by Cozzens [8] that computing the boxicity of a graph is NP–hard. That is why we pursue the problem of finding the boxicity for certain classes of graphs.

As described in Subsection 3 and Subsection 4, we have obtained polynomial-time algorithms for computing the boxicity of the complements of trees and the complements of line graphs of trees.

Given that trees are outer planar graphs, we can probe for a more general result. Now that we have an algorithm for computing the boxicity of the complement of a tree, it might be possible to derive an algorithm for computing the boxicity of the complement of an outerplanar graph. Note that the boxicity of outerplanar graphs has been proved by Scheinerman [19] to be most 2.

Further work on proving that the boxicity of the Mycielskian of a cycle, $\mathrm{M}(C_k)$, was equal to 3 for all $k \geq 5$ would also be considered a productive endeavour. In the interest of time, we left some potential avenues unexplored, and we would like to continue this work.

References

[1] A. Adiga, D. Bhowmick, and L. S. Chandran. Boxicity and poset dimension. *SIAM Journal on Discrete Mathematics*, 25(4):1687–1698, 2011.

[2] M. Caoduro. *Geometric challenges in combinatorial optimization : packing, hitting, and coloring rectangles.* Theses, Université Grenoble Alpes [2020-....], Nov. 2022.

[3] M. Caoduro and L. Lichev. On the boxicity of kneser graphs and complements of line graphs. *Discrete Mathematics*, 346(5):113333, 2023.

[4] M. Caoduro and A. Sebő. Boxicity and interval-orders: Petersen and the complements of line graphs, 2023.

[5] L. S. Chandran, A. Das, and C. Shah. Cubicity, boxicity and vertex cover, 2007.

[6] L. S. Chandran, M. C. Francis, and N. Sivadasan. Boxicity and maximum degree. *Journal of Combinatorial Theory, Series B*, 98(2):443–445, 2008.

[7] L. S. Chandran and N. Sivadasan. Boxicity and treewidth. *Journal of Combinatorial Theory, Series B*, 97(5):733–744, 2007.

[8] M. B. Cozzens. *Higher and multidimensional analogues of interval graphs.* Theses, Rutgers University, New Brunswick, 1981.

[9] M. B. Cozzens and F. S. Roberts. Computing the boxicity of a graph by covering its complement by cointerval graphs. *Discrete Applied Mathematics*, 6(3):217–228, 1983.

[10] L. Esperet. Boxicity of graphs with bounded degree. *European Journal of Combinatorics*, 30(5):1277–1280, 2009.

[11] L. Esperet and G. Joret. Boxicity of graphs on surfaces. *Graphs and combinatorics*, 29(3):417–427, 2013.

[12] A. Kamibeppu. Bounds for the boxicity of mycielski graphs, 2015.

[13] M. Kneser. Aufgabe 360. *Jahresbericht der Deutschen Mathematiker-Vereinigung 2*, 27, 1955.

[14] J. Mycielski. Sur le coloriage des graphs. *Colloquium Mathematicum*, 3:161–162, 1955.

[15] R. J. Opsut, F. S. Roberts, et al. On the fleet maintenance, mobile radio frequency, task assignment, and traffic phasing problems. *The theory and applications of graphs*, pages 479–492, 1981.

[16] F. Roberts. *Food webs, competition graphs, and the boxicity of ecological phase space*, pages 477–490. 11 2006.

[17] F. S. Roberts. On the boxicity and cubicity of a graph. *Recent progress in combinatorics*, 1:301–310, 1969.

[18] F. S. Roberts. Discrete mathematical models, with applications to social, biological, and environmental problems. 1976.

[19] E. R. Scheinerman. *Intersection classes and multiple intersection parameters.* Theses, Princeton University, 1984.

[20] A. Scott and D. Wood. Better bounds for poset dimension and boxicity. *Transactions of the American Mathematical Society*, 373(3):2157–2172, 2020.

Appendix A. Code

```
def greedyVertices(liveVertices, usedVertices):
    remainingVertices = set(kgc.vertices()).difference(set(usedVertices))
    tempNewLiveVertices = [(v, liveVertices.intersection(KGC_NEIGHBORS[frozenset(v)])) for v in remainingVertices]
    maxDegree = max([len(templist) for v, templist in tempNewLiveVertices])
    maxVertices = [v for v, tempList in tempNewLiveVertices if len(tempList) == maxDegree]
    return maxVertices
```

Listing 1. The Greedy Algorithm