



Virginia Commonwealth University
VCU Scholars Compass

Theses and Dissertations


Graduate School

2016

Automated Conjecturing Approach for Benzenoids

David Muncy
Virginia Commonwealth University

Follow this and additional works at: <https://scholarscompass.vcu.edu/etd>

 Part of the [Artificial Intelligence and Robotics Commons](#), and the [Discrete Mathematics and Combinatorics Commons](#)

© The Author

Downloaded from

<https://scholarscompass.vcu.edu/etd/4608>

This Thesis is brought to you for free and open access by the Graduate School at VCU Scholars Compass. It has been accepted for inclusion in Theses and Dissertations by an authorized administrator of VCU Scholars Compass. For more information, please contact libcompass@vcu.edu.

AUTOMATED CONJECTURING APPROACH FOR BENZENOIDS

A thesis submitted in partial fulfillment of the requirements for the degree of Master of Science at Virginia Commonwealth University.

by

David Muncy
Master of Science

Director: C. E. Larson, Associate Professor
Department of Mathematics and Applied Mathematics

Committee Member: J. P. Brooks, Associate Professor
Department of Statistics and Operations Research

Committee Member: R. H. Hammack, Professor
Department of Mathematics and Applied Mathematics



Virginia Commonwealth University
Richmond, Virginia
December 2016

Table of Contents

Abstract	iv
1 Invariant investigation	1
1.1 Embeddings	1
1.2 Domination number and dominating sets	5
1.3 Invariants	9
1.4 Methods of proof	23
1.5 Results	29
1.5.1 General benzenoids	31
1.5.2 Catacondensed benzenoids	36
1.6 More conjectures	43
2 Property probe	45
2.1 Initial properties	48
2.2 The conjecture-making process	51
2.3 More properties	62
2.4 Clustering	68
2.5 Results	75
2.6 Open conjectures	83
Bibliography	85

A Invariant and property code

Abstract

Benzenoids are graphs representing the carbon structure of molecules, defined by a closed path in the hexagonal lattice. These compounds are of interest to chemists studying existing and potential carbon structures. The goal of this study is to conjecture and prove relations between graph theoretic properties among benzenoids. First, we generate conjectures on upper bounds for the domination number in benzenoids using invariant-defined functions. This work is an extension of ideas to be presented in C. E. Larson, et. al. [7]. Next, we generate conjectures using property-defined functions. As the title indicates, the conjectures we prove are not thought of on our own, rather generated by a process of automated conjecture-making. This program, named CONJECTURING¹, is developed by Craig Larson² and Nico Van Cleemput³.

¹<http://nvcleemp.github.io/conjecturing/>

²Department of Mathematics and Applied Mathematics, Virginia Commonwealth University, Richmond, VA 23284, clarson@vcu.edu

³Applied Mathematics and Computer Science, Ghent University, Krijgslaan 281 S9, 9000 Ghent, Belgium, nico.vancleemput@gmail.com

Chapter 1

Invariant investigation

Our motivation for studying benzenoid hydrocarbons, *benzenoids* for short, stems from research on the chemical properties of carbon chains. Initially, ideas conjectured by Müller and Müller-Rodloff in [12] motivated a topological investigation of benzenoids with stability results recorded by Longuet-Higgins in [10] showing certain benzenoid structures have a low chemical stability. S. J. Cyvin and I. Gutman note in [3] of this topological development and later define algebraic relations on graphs of benzenoids embedded in the plane in their book *Introduction to the Theory of Benzenoid Hydrocarbons* [5]. This is a study of applying graph theory by defining graph invariants from benzenoids for interpretation on the respective chemical structure. We begin by discussing benzenoid embeddings to explain the graph theory used in proving automated conjectures on upper bounds of the domination number of a benzenoid, denoted $\gamma(B)$. These bounds, and other results of this chapter, are a joint work with Laura Hutchinson, Vikram Kamat, Craig Larson, Shailya Mehta, and Nico Van Cleemput.

1.1 Embeddings

A benzenoid, denoted B , may be drawn as a graph defined with a vertex set and an edge set. Benzene is the benzenoid with six vertices and six edges. These edges connect

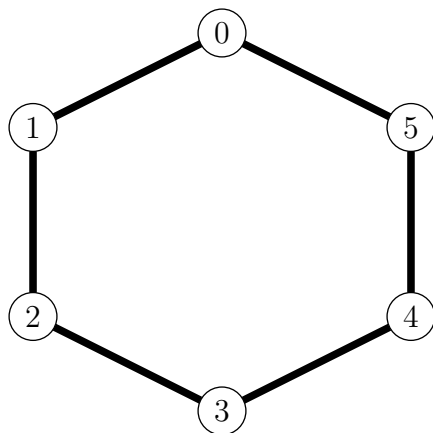


Figure 1.1: Benzene graphed as C_6

the vertices in a closed loop in which neither a vertex nor edge is repeated. This type of closed loop, in which no vertex or edge is repeated, is defined as a *cycle*. A cycle graph of order n is a graph with n vertices consisting of a single cycle. Cycle graphs are denoted C_n . Benzene is equivalent to cycle graph C_6 . The graph of benzene in Figure 1.1 has vertex set $V(C_6) = \{0, 1, 2, 3, 4, 5\}$ and edge set $E(C_6) = \{01, 12, 23, 34, 45, 50\}$.

An arbitrary benzenoid is constructed by defining a closed curve on the hexagonal lattice, shown in Figure 1.2. The closed curve that forms the benzenoid is called the *perimeter*. We define benzenoids by carving them out of the hexagonal lattice. The act of slicing the hexagonal lattice defines the vertex and edge set of a benzenoid. Benzene is the graph represented as a single hexagon from the lattice. Structuring benzenoids as hexagonal rings is practical and allows for graph theoretic interpretation and computation. A benzenoid formed from a closed curve is a *nonseparable* graph. Figure 1.3 shows three more benzenoids embedded in the plane.

Definition 1. A graph G is nonseparable if, for any $v \in V(G)$, the graph $G - v$ is connected.

One analogy used to visualize a benzenoid is to consider one comprised of *blocks* forming a nontrivial connected graph G . The individual blocks combine to form a *cluster*, which is the full picture of a benzenoid. The definition for a block is imported from Chartrand, Lesniak, & Zhang [2]; cluster is introduced here.

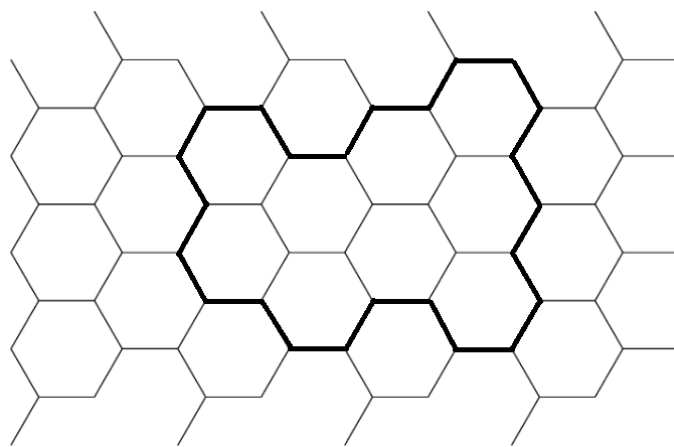
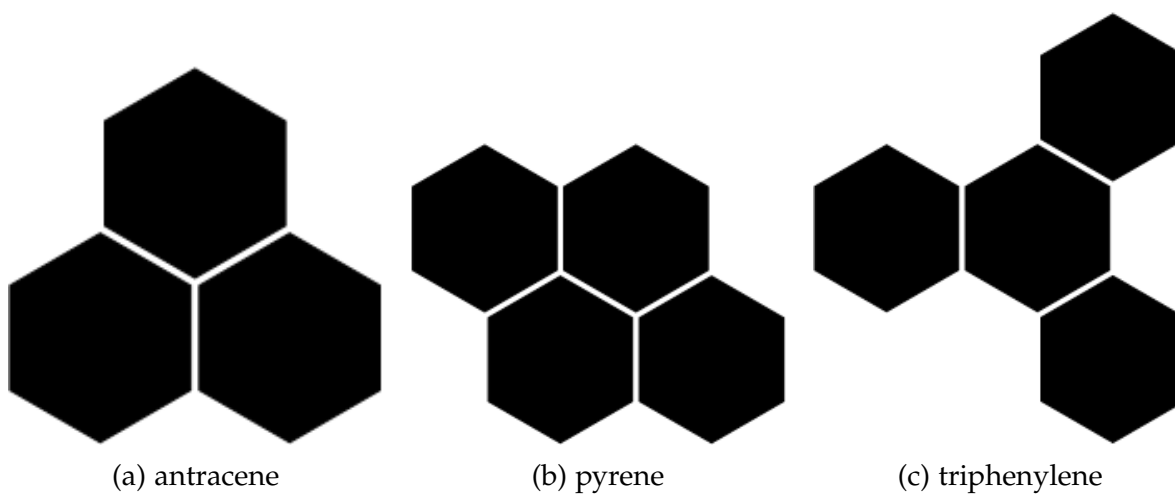


Figure 1.2: Defining a benzenoid by closing a curve in the hexagonal lattice



(a) anthracene

(b) pyrene

(c) triphenylene

Figure 1.3: Three benzenoids

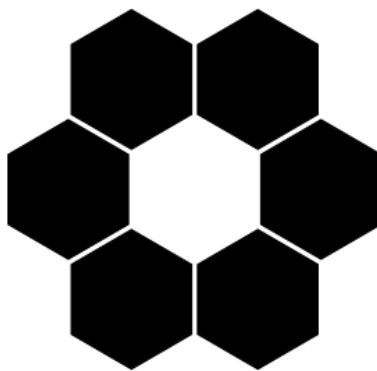


Figure 1.4: Methylcoroene is not a benzenoid as there are six edges that can form a hexagon in the center, but a hexagon is not present.

Definition 2. A block of nontrivial connected graph G is a maximal nonseparable subgraph of G .

Definition 3. A cluster is a nonseparable subgraph of the hexagonal lattice, not necessarily maximal.

Benzenoids have three chemical properties that are not represented in an embedding. First, hydrogen atoms bonded to carbon atoms are not shown. Hydrogen atoms are assumed to be bonded to carbon atoms on the perimeter. Second, double and single bonds in benzenoids are not distinguished in an embedding. Each edge is single sided. Finally, embeddings do not account for π -electrons that may exist in double bounds of a benzenoid.

In addition, there are guidelines to follow when constructing a benzenoid. First, is the *slicing rule*, which means that we may define benzenoids by peeling them for the hexagonal lattice. This ensures the absence of holes in an embedding and is analogous defining benzenoids by forming closed curves. Any possible hexagon that may form where six edges form a cycle must be present in an embedding. An empty space flanked by six hexagons is not allowed. For example, the graph of methylcoroene shown in Figure 1.4 is not an embedding of a benzenoid—it does not follow the slicing rule as there is hexagon missing where six edges form a cycle.

Methylcoroene is from a larger class of hydrocarbons known as polycyclic aromatic

hydrocarbons (PAHs). Benzenoids are PAHs that are constructed from benzene rings. Gutman & Cyvin in [5] distinguish the PAH class from the benzenoid subclass by explaining that PAHs may be, and often are, formed cycles with fewer than six carbon atoms. In addition, many PAHs have other carbon activity that the hexagonal embedding does not account for. They later note that most PAHs are thermodynamically and chemically stable. Comparatively, the reactive nature of some benzenoids allow for varying chemical stability dependent on carbon structure. Vertex positioning in the embedding of benzenoids may be significant in determining chemical stability. One measure of efficient vertex placement is given by the *domination number* of a benzenoid, which denotes how many vertices are needed to dominate the vertex set. Bounding the domination number of benzenoids is the primary motivation of this invariant study.

1.2 Domination number and dominating sets

For an arbitrary graph G , a vertex $v \in V(G)$ is said to *dominate* both itself and any vertices adjacent to v . That is, a vertex v dominates the vertices in the closed neighborhood $N[v]$. So vertex v and vertices sharing an edge with v are dominated by v . Understanding the concept of domination is the first step to forming *dominating sets* of vertices from $V(G)$.

Definition 4. A set S of vertices of graph G is a dominating set of G , if every vertex of G is dominated by at least one vertex of S .

There is at least one dominating set for any arbitrary graph G , the trivial dominating set S containing every element of $V(G)$. As with many mathematical objects, we are not interested in an arbitrary dominating set, but rather a *minimum* dominating set. A minimum dominating set of G is a dominating set of G containing the fewest number of elements. The *domination number* of a graph G , denoted $\gamma(G)$, is described by Chartrand & Lesniak in [2] as the minimum cardinality among all dominating sets. This invariant reports the fewest number of vertices needed to dominate every vertex in the vertex

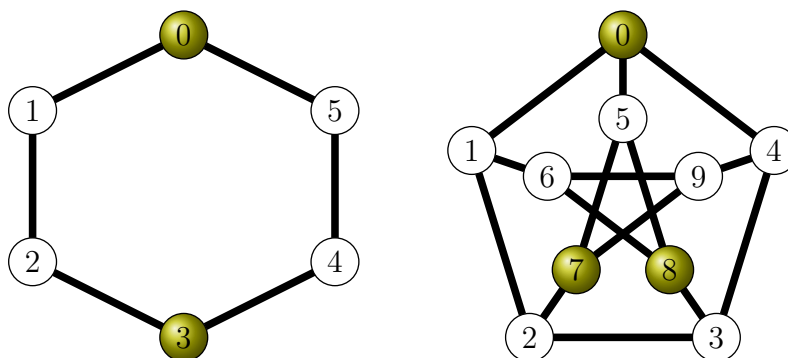


Figure 1.5: A minimum dominating set of benzene: $\{0, 3\}$, the Petersen Graph: $\{0, 7, 8\}$

set. The domination number of a graph is the basis of this invariant investigation and warrants a formal definition.

Definition 5. *The domination number $\gamma(G)$ of a graph G is the cardinality of a minimum dominating set.*

Again, this number counts fewest number of nodes in a dominating set so that these nodes are adjacent to all other nodes. These dominating nodes are vertices from the dominating set of a graph, which is a subset of the vertex set of a graph.

Remark. *Let $D_{\min}(B)$ denote a minimum dominating set of benzenoid B .*

Benzene has a minimum dominating set from choosing two vertices in $V(C_6)$ which will not share an edge in $E(C_6)$. It is possible to form three minimum dominating sets of benzene. Let $D_{\min}(C_6)$ denote a minimum dominating set of benzene. Once we select a vertex to be the first in the minimum dominating set, the remaining vertex must be the one that shares no edge with a vertex already dominated by the vertex chosen for $D_{\min}(B)$. One dominating set of benzene is shown in Figure 1.5 as well as a dominating set of the Petersen graph. Gold vertices in the figure belong to the minimum dominating set. From the labeling of Figure 1.1, the three minimum dominating sets of benzene are

$$\{0, 3\}, \quad \{1, 4\}, \quad \text{and} \quad \{2, 5\}.$$

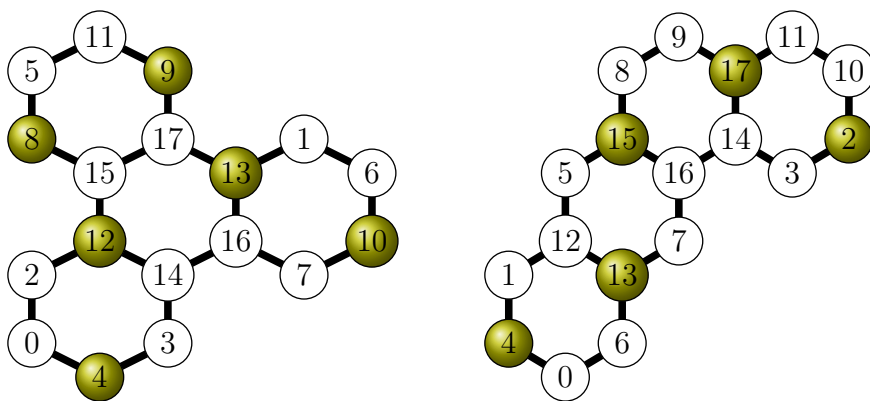


Figure 1.6: Example of overlap: let B be Triphenylene with $n(B) = 18$, $h(B) = 4$, $\gamma(B) = 6$ and B' be Tetraphene with $n(B') = 18$, $h(B') = 4$, and $\gamma(B') = 5$

In the best case, for a benzenoid B with $h(B)$ hexagons and order $n(B)$, each dominating vertex in $D_{\min}(B)$ dominates at most three other vertices uniquely. This shows that for any benzenoid the domination number $\gamma(B)$ has a lower bound of $\frac{n(B)}{4}$. One of every four vertices appear in the dominating set. Some benzenoids are less efficient than others at spreading their dominating vertices. When a vertex $v \in D_{\min}(B)$ is adjacent to a vertex already dominated by a different vertex $u \in D_{\min}(B)$, the minimum dominating set is said to have *overlap*. For a benzenoid with every minimum dominating set containing overlap directly affects the domination number. Consider a different benzenoid B' of the same order and number of hexagons as B , but has a minimum dominating set without overlap. Necessarily, $\gamma(B') \leq \gamma(B)$ and the cardinality of $D_{\min}(B)$ is larger than or equal to the cardinality of $D_{\min}(B')$. Refer to Figure 1.6 for an example of overlap through a side-by-side comparison of the dominating sets of Triphenylene and Tetraphene. From the figure we see that, although Triphenylene and Tetraphene have the same order and number of hexagons, a minimum dominating set of Tetraphene contains fewer elements than a minimum dominating set of Triphenylene. In other words, this minimum dominating set of Triphenylene contains overlap. It turns out that all minimum dominating sets of Triphenylene have overlap as well.

A large number of dominating vertices in B may be interpreted as a strain on the vertices to reach every element of $V(B)$. One idea is that a larger spacing of carbon

atoms causes a strain by introducing more π -electrons than another benzenoid of equal order with *compact* structure, meaning it has a smaller domination number.

A dominating set may be quickly checked to be minimum if provided, although an efficient way to find a minimum dominating set is not known. In computer science, this problem of not having an efficient algorithm to generate a solution is indicated as an *NP-complete* decision problem. This term refers the problem of determining minimum dominating sets of graphs not having a known solution in polynomial time. Instead, finding minimum dominating sets of graphs is solved by current algorithms in exponential time. See Section 3.2 in [2] for an introduction to P and NP decision problems in graph theory. Calculating the domination number for an arbitrary graph is known to be NP-complete from Garey & Johnson [4].

It is useful to use properties found in all benzenoid graphs as a filter separating out arbitrary graphs for ones with specific structure. Benzenoids belong to the class of graphs that are *bipartite*. The characterization of bipartite graphs are that their vertex sets can be separated into two independent sets of nonadjacent vertices. Another class benzenoids belong to are planar graphs. A planar graph may be embedded in the plane without any edge crossings. Benzenoids are constructed by defining a closed path on the hexagonal lattice and thus do not contain edge crossings. Determining minimum dominating sets in planar graphs and bipartite graphs are additional known NP-complete decision problems from [4] and M. Liedloff [9]. Benzenoids have more properties which further restrict the graphs belonging to its class. One restriction found in all benzenoids are on the *degree* of vertices in a vertex set. The following definition appears in [2].

Definition 6. For a graph G , the degree of a vertex v in G is the number of vertices in G that are adjacent to v .

For an arbitrary benzenoid B , any $v \in V(B)$ has a degree of 2 or 3. Similar to membership in planar graphs, this is due to benzenoids being connected by closed curves on the hexagonal lattice. Defining a closed curve introduces internal vertices and external

vertices. Internal vertices have a degree of 3, and external vertices (internal in the infinite hexagonal lattice) have a degree of 2. From this observation it must be that any vertex $v \in D_{\min}(B)$ may have a maximum degree of at most 3. However, this extra specification on vertex degrees does not make finding minimum dominating sets easier. A result by T. Kikuno, N. Yoshida, and Y. Kakuda in [6] shows that determining minimum dominating sets in planar graphs with maximum degree of 3 is NP-complete. Still, benzenoids possess even more structure that may play a significant role in determining an algorithm to find minimum dominating sets in polynomial time. It remains an open question if finding minimum dominating sets of a benzenoid is NP-complete.

In the face of not knowing an efficient algorithm for finding minimum dominating sets it is useful to determine bounds on a benzenoid's domination number. Upper and lower bounds narrow the interval containing the domination number. Narrowing the interval becomes the goal. Defining more invariants in addition to the domination number gives CONJECTURING more bounds to work with. Strengthening an upper bound on $\gamma(B)$ is our primary motivation for defining more graph invariants for benzenoids. We use a systematic approach for generating the upper bounds on $\gamma(B)$ by deploying CONJECTURING. The CONJECTURING program serves as a tool to help us refine conjectures by relating other defined graph invariants to the domination number in combinations we may have glossed over. As the list of benzenoids and invariant-defined functions supplied to CONJECTURING grows, so does the program's generated output.

1.3 Invariants

Graph invariants are split into two categories for this study. The first group of invariants may be calculated for any arbitrary graph. The second group of invariants are those for which it only makes sense to talk about in the context of benzenoids. In this group, invariants are defined specifically for the benzenoid class of graphs. First we

will discuss the invariant-defined functions that apply for arbitrary graphs. Defining an invariant is synonymous with programming an invariant-defined function. All of the invariants below are programmed in SageMath, the back-end for running CONJECTURING. The conjecture-making process is detailed in Section 2.2 and code for each invariant is included in Appendix A. Using CONJECTURING, SageMath, and the provided code it is possible to reproduce the study for individual experimentation.

Graphs are defined by a vertex set and edge set, and the first two invariants often determined for a graph are its order and size. The order of a graph is the cardinality of its vertex set and the size of a graph is the cardinality of its edge set. These are the first invariants we define for benzenoids.

Invariant 1 (Number of vertices). *For a benzenoid B , define $B.order()$ as the number of vertices of B .*

Invariant 2 (Number of edges). *For a benzenoid B , define $B.size()$ as the number of edges of B .*

These invariants are programmed with the names $B.order()$ and $B.size()$. We shorten the notation by referring to the two invariants as $n(B)$ and $m(B)$ respectively. Next, we program an invariant-defined function to determine a benzenoid's domination number. This is our invariant of interest for this investigation and is formally stated as follows.

Invariant 3 (Domination number). *For a benzenoid B , define $domination_number(B)$ to be the cardinality of a minimum dominating set of B .*

In this study we denote the domination number of a benzenoid as $\gamma(B)$ in compliance with the established notation found in graph theory texts.

We observed in the previous section that vertices in a benzenoid may have a degree of 2 or 3. The number of degree-2 and degree-3 vertices in a benzenoid could help bound its domination number, so these two invariants are worth defining. We can define these

invariants by looking at degree sequences of benzenoids and counting the amount of times 2 and 3 appears in the sequence.

Invariant 4 (Number of degree-2 vertices). *For a benzenoid B , define $degree_2_vertices(B)$ as the number of degree-2 vertices in B .*

Invariant 5 (Number of degree-3 vertices). *For a benzenoid B , define $degree_3_vertices(B)$ as the number of degree-3 vertices in B .*

For simplification, denote the number of degree-2 vertices in B as $\delta_2(B)$ and the number of degree-3 vertices in B as $\delta_3(B)$. Using a shorter notation is more convenient when writing and proving propositions containing many invariants.

Using this same mindset we also want to determine the number of different edge types found in a benzenoid. By construction, for any benzenoid, an edge may have endpoints with two degree-2 vertices, two degree-3 vertices, or one degree-2 vertex and one degree-3 vertex. The varying amounts of these three edge types are something worth accounting for. These are three more graph invariants to define.

Invariant 6 (Number of 2-2 edges). *For a benzenoid B , define $number_of_2_2_edges(B)$ as the number of edges in B that are connected by two degree-2 vertices.*

Invariant 7 (Number of 2-3 edges). *For a benzenoid B , define $number_of_2_3_edges(B)$ as the number of edges in B that are connected by a degree-2 vertex and a degree-3 vertex.*

Invariant 8 (Number of 3-3 edges). *For a benzenoid B , define $number_of_3_3_edges(B)$ as the number of edges in B that are connected by two degree-3 vertices.*

We denote these three invariants as $e_{22}(B)$, $e_{23}(B)$, and $e_{33}(B)$, respectively. Computing these three numbers turns out to be more involved than the previous invariants as SageMath does not have a built-in method for the calculations. Programming the invariants is left to us. As shown in Appendix A, we determine these invariants by initializing

a counter (at 0) and defining two endpoints for each edge. We denote these two endpoints as `endpoint1` and `endpoint2`. Then the degree is calculated for each endpoint. Using an `if-then` statement SageMath checks each edge and determines if the edge's endpoints have the degrees we are seeking. If so, the counter is increased by 1 and the total is returned after all edges are checked. For instance, if we wanted to determine the number of 2-2 edges in B , we specify the `if` statement such that if both `endpoint1` and `endpoint2` are degree-2, then add 1 to the counter. In this way we filter through the endpoints and only count the edges joined by two degree-2 endpoints. The invariants $e_{23}(B)$ and $e_{33}(B)$ are found in a similar fashion.

The final invariant-defined function applicable in an arbitrary graph is the graph's *matching number*. This invariant counts the number of elements in a graph's maximum *independent edge set*. Both terms are formally defined below, followed by a statement of the invariant.

Definition 7. For a graph G an independent edge set of G is a subset $S \subseteq E(G)$ such that no two edges in S share a vertex in $V(G)$.

Definition 8. For a graph G , the matching number of G is the cardinality of a maximum independent edge set of G .

Invariant 9 (Matching number). For a benzenoid B , define $matching_number(B)$ to be the cardinality of a maximum independent edge set of B .

SageMath has built-in functionality for finding a maximum independent edge set for a graph G ; the method is programmed as `G.matching()`. Calculating the length of this set is an easy way to define the matching number.

Next we move to defining invariants that do not necessarily apply to an arbitrary graph, rather ones in a specific class. These next few invariants are all about a benzenoid's two bipartite sets. As mentioned in the previous section, benzenoids are known

bipartite graphs. Before stating the invariants we introduce the notation for the *smaller bipartite set* and the *larger bipartite set*.

Remark. Let $\mathcal{B}_s(B)$ and $\mathcal{B}_l(B)$ denote the smaller and larger bipartite sets, respectively, of an arbitrary benzenoid B .

The smaller and larger bipartite sets are just that: sets; they are not numbers and therefore not invariants. However, we may define invariants based on the cardinality of these sets.

Invariant 10 (Cardinality of the smaller bipartite set). For a benzenoid B , define *smaller_bipartite_set*(B) as the number of elements in the smaller partite set of B .

Invariant 11 (Cardinality of the larger bipartite set). For a benzenoid B , define *larger_bipartite_set*(B) as the number of elements in the larger partite set of B .

The notation for Invariant 10 is $|\mathcal{B}_s(B)|$ and for Invariant 11 is $|\mathcal{B}_l(B)|$. When the two invariants are equal (that is, when $|\mathcal{B}_s(B)| = |\mathcal{B}_l(B)|$) each set has the same number of elements. In this case, we say that benzenoid B has *balance*. This is a property, not a number. For any benzenoid we can determine if the property is True or False. Either the benzenoid has the property *balance* or it does not. Property conjectures are examined in Chapter 2 of this paper, an extension of this study on invariants. When a benzenoid does not have balance, it must be that $|\mathcal{B}_l(B)| > |\mathcal{B}_s(B)|$. This property is denoted *color excess*. The amount of color excess, denoted $\triangleleft(B)$, varies between benzenoids and gives us another invariant to define on the bipartite sets of B .

Invariant 12 (Color excess). For a benzenoid B , define *color_excess*(B) as the difference between the cardinality of the larger bipartite set of B and the cardinality of the smaller bipartite set of B .

The next invariant-defined function counts the number of hexagons in an embedding of a benzenoid.

Invariant 13 (Number of hexagons). *For a benzenoid B , define $\text{hexagons}(B)$ as the number of hexagons the embedding of B .*

The notation $h(B)$ is used to denote Invariant 13. While it is apparent this is a useful invariant, it is not immediately clear how to go about defining it. There is not a built-in method in SageMath which will recognize and count each hexagon in an embedding and return the total amount. This is the case for most categorically benzenoid invariants. We need to come up with our own method. Many of these invariants already have defined relations, appearing in Gutman & Cyvin's book [5]. The formula for the number of hexagons, as well as other invariants, may be imported from the book into SageMath for use in CONJECTURING. Along the way we will prove some of these relations, starting with $h(B)$.

Many of the formulas for invariants and automated conjecture results rely on an assumption that every non-trivial benzenoid, i.e. not benzene, has a *removable hexagon*. A removable hexagon is a hexagon H in benzenoid B where, if we remove the vertices and edges in $V(B)$ and $E(B)$ that are unique to H , the resulting graph is a benzenoid. The benzenoid formed from removing H is often denoted B' . In the induction step of many proofs we often use the following removable hexagon lemma to guarantee the existence of a benzenoid after the removal of a removable hexagon. This lemma makes use of a benzenoid's *inner dual* which we define first. Examples and applications of inner duals and appear in Section 1.4.

Definition 9. *The inner dual of a benzenoid B is a graph that has a vertex for each face of B , excluding the outer face. This graph contains an edge between two vertices whenever two hexagons in B share an edge.*

Lemma 1. *Every non-trivial benzenoid B has a removable hexagon.*

Proof. Let H_1 and H_2 be hexagons in B of maximum distance from the diameter of the corresponding inner dual. We claim that H_1 is removable. Suppose not. Then $B - H_1$

is disconnected. Let C_1 and C_2 be two components of $B - H_1$ with H_2 contained in C_1 . Then any path of hexagons from H_2 to C_2 must contain H_1 . Let H_3 be any hexagon in C_2 . Then the distance from H_2 to H_3 is greater than the distance from H_2 to H_1 . This is a contradiction to the assumption that H_1 and H_2 were at maximum distance. Then we were wrong to assume that H_1 is not removable. It follows by contradiction that every non-trivial benzenoid has a removable hexagon. \square

Theorem 1. *Suppose benzenoid B is of order $n(B)$ and size $m(B)$. Then the number of hexagons in B , denoted $h(B)$, is given by $h(B) = m(B) - n(B) + 1$.*

Proof. Let B be a benzenoid and let H be a removable hexagon of B . By Lemma 1 we know at least one removable hexagon must exist in B . Let B' be the benzenoid found by removing the edge(s) in H not shared by any other removable hexagon of B . By assumption, we have that $h(B') = m(B') - n(B') + 1$ and $h(B) = h(B') + 1$. Suppose removing H removes m' edges from $E(B)$ such that $m(B') = m(B) - m'$. Then $m' - 1$ vertices are removed, and $n(B') = n(B) - (m' - 1)$. Therefore,

$$\begin{aligned}
 h(B) &= h(B') + 1 \\
 &= m(B') - n(B') + 1 + 1 \\
 &= (m(B) - m') - (n(B) - m' + 1) + 1 + 1 \\
 &= m(B) - n(B) + 1.
 \end{aligned}$$

\square

When programming this invariant in SageMath we may take advantage of knowing the formula by defining `hexagons(B)` to return the quantity $m(B) - n(B) + 1$.

Let's return to the discussion of vertices and edges. The amount of vertices and amount of edges in a vertex and edge set are typically the first invariants determined when given a graph. Earlier, we split edges into three types based on the degrees of their

endpoints. In a similar manner we may categorize vertices into two groups: *internal* vertices and *external* vertices. An external vertex is one that lies on the perimeter of a benzenoid. Recall the term perimeter from Section 1.1. We simply mean that an external vertex of a benzenoid lies on the closed curve defined on the hexagonal lattice forming the benzenoid. Internal vertices are the vertices that are not external; they do not lie on the closed curve forming the benzenoid. Every benzenoid has external vertices, but having internal vertices is not a requirement. The amount of internal and external vertices in a benzenoid constitutes the next two invariants.

Invariant 14 (Number of internal vertices). *For a benzenoid B , define $number_of_internal_vertices(B)$ as the number of internal vertices in B .*

Invariant 15 (Number of external vertices). *For a benzenoid B , define $number_of_external_vertices(B)$ as the number of external vertices in B .*

Let $n_i(B)$ denote the number of internal vertices in B and $n_e(B)$ denote the number of external vertices in B . Necessarily, internal vertices of a benzenoid have a degree of 3. However, external vertices may have a degree of 2 or 3. The point is it would be wrong to assume a vertex is internal or external just from its degree. These invariants may be, and usually are, different than $\delta_2(B)$ and $\delta_3(B)$. A formula for the number of internal vertices appears in [5], and is proven in the following theorem.

Theorem 2. *Let B be a benzenoid with $h(B)$ hexagons and $n(B)$ vertices. Then the number of internal vertices in B , denoted $n_i(B)$, is given by $4h(B) + 2 - n(B)$.*

Proof. Using Theorem 1, we want to show that

$$\begin{aligned} n_i(B) &= 4h(B) + 2 - n(B) \\ &= 4(m(B) - n(B) + 1) + 2 - n(B) \\ &= 4m(B) - 5n(B) + 6. (*) \end{aligned}$$

Use mathematical induction on $h(B)$. In the base case when $h(B) = 1$, benzenoid B is benzene which has 0 internal vertices. Notice that $n_i(B) = 0 = 4(1) + 2 - 6 = 4h(B) + 2 - n(B)$, so the base case holds.

Suppose that the assumption that $n_i(B) = 4h(B) + 2 - n(B)$ holds for benzenoids with $k - 1 \geq 1$ hexagons for $k \in \mathbb{N}$. We will show the assumption holds when $h(B) = k$. Let B be a benzenoid with $h(B) = k$. By Lemma 1, there exists a removable hexagon H in B . Let $B' = B - H$ be the benzenoid defined by the removal of H from B . Removing this hexagon defines the following relations:

1. The number of hexagons in B' is given by $h(B') = h(B) - 1 = k - 1$.
2. Removing H from B removes m' edges in B , returning the relation $m(B') = m(B) - m'$.
3. Since m' edges are removed from B , $m' - 1$ vertices are removed from B as well, showing that $n(B') = n(B) - (m' - 1)$.

As $h(B') = k - 1$ we may apply the inductive hypothesis. Observe that

$$\begin{aligned}
 n_i(B') &= 4h(B') + 2 - n(B') \\
 &= 4(m(B') - n(B') + 1) + 2 - n(B') \\
 &= 4m(B') - 5n(B') + 6 \\
 &= 4(m(B) - m') - 5(n(B) - m' + 1) + 6 \\
 &= 4m(B) - 5n(B) + m' + 1. (**)
 \end{aligned}$$

What remains to be shown is that equations (*) and (**) are equivalent. That is, we want to show

$$\begin{aligned}
 n_i(B') - m' - 1 &= n_i(B) - 6 \\
 n_i(B') - m' + 5 &= n_i(B).
 \end{aligned}$$

We proceed by cases. First suppose that $m' = 1$, meaning 1 edge was eliminated from

B by the removal of H. When 1 edge is removed, the vertices lying on the perimeter in B' where H was removed switch from internal to external vertices. There are 4 vertices that external in B' and internal in B. Then the equation determining the number of internal vertices in benzenoid B' is given by $n_i(B') = n_i(B) - 4$. From the relation above

$$\begin{aligned} n_i(B) &= n_i(B') - m' + 5 \\ &= n_i(B) - 4 - 1 + 5 \\ &= n_i(B), \end{aligned}$$

as we wanted to show. The remaining cases follow in a similar manner.

Now suppose that $m' = 2$. Then 2 edges are removed from $E(B)$ by the removal of H. Removing 2 edges removes 1 vertex from the perimeter of B, and now 3 more vertices lie on the perimeter of B' , with 5 total external vertices on the perimeter where H once was. Removing $m' = 2$ edges creates 3 external vertices, so $n_i(B') = n_i(B) - 3$. Then

$$\begin{aligned} n_i(B) &= n_i(B') - m' + 5 \\ &= n_i(B) - 3 - 2 + 5 \\ &= n_i(B). \end{aligned}$$

Suppose that $m' = 3$. Removing 3 edges from $E(B)$ removes 2 external vertices on the perimeter of B. Then 4 vertices in B' lie on the perimeter where H was removed. So 2 vertices that were internal in B are external in B' , giving the relation $n_i(B') = n_i(B) - 2$. Now we find that

$$\begin{aligned} n_i(B) &= n_i(B') - m' + 5 \\ &= n_i(B) - 2 - 3 + 5 \\ &= n_i(B). \end{aligned}$$

Consider when $m' = 4$. Removing 4 edges from $E(B)$ also removes 3 external vertices lying the perimeter of B , leaving 3 external vertices on the perimeter of B' where H was removed. There is 1 internal vertex in B that is now external in B' . This gives the relation $n_i(B') = n_i(B) - 1$, and we determine that

$$\begin{aligned} n_i(B) &= n_i(B') - m' + 5 \\ &= n_i(B) - 1 - 4 + 5 \\ &= n_i(B). \end{aligned}$$

In the final case $m' = 5$. Removing 5 edges from $E(B)$ removes 4 external vertices from $V(B)$. Since 5 edges are removed, it must be that H shares only 1 edge with another hexagon. Then the 2 vertices in B' that remain from the removal of H are external vertices, and we find that $n_i(B') = n_i(B) - 0$. Then the relation we want to show is now

$$\begin{aligned} n_i(B) &= n_i(B') - m' + 5 \\ &= n_i(B) - 0 - 5 + 5 \\ &= n_i(B). \end{aligned}$$

In all cases, we are able to show that $n_i(B') - m' + 5 = n_i(B)$. It follows by mathematical induction that the number of internal vertices in B is given by $n_i(B) = 4h(B) + 2 - n(B)$. □

Corollary 1. *Let B be a benzenoid with $h(B)$ hexagons and $n(B)$ vertices. The number of external vertices in B is equal to the order of B minus the number of internal vertices of B .*

Proof. A vertex $v \in V(B)$ may be placed in one of two disjoint sets dependent on if v is an internal vertex or external vertex. The order of B may be calculated by summing the number of internal vertices and the number of external vertices. That is, $n(B) = n_i(B) + n_e(B)$. Subtracting both sides of the equation by $n_i(B)$ proves the statement. □

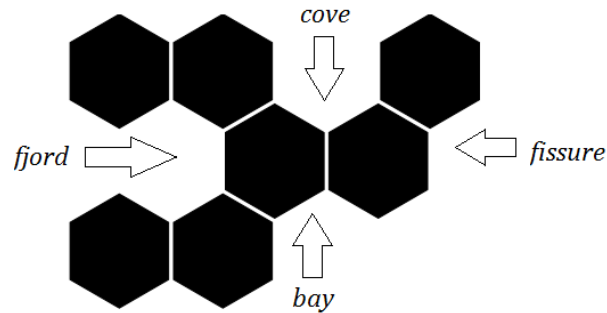


Figure 1.7: A benzenoid containing all four possible bay regions

Edges can be classified as external or internal as well. Defining usable formulas first requires an understanding of *bay regions* in a benzenoid. Bay regions are structural features that exist on the perimeter of a benzenoid. There are four possible bay regions that may appear in a benzenoid: *fissures*, *bays*, *coves* and *fjords*. The meaning of each of these terms is best explained visually, and each bay region is shown in Figure 1.7.

The number of bay regions, as well as the number of each type of bay regions, form the next four invariants.

Invariant 16 (Number of bay regions). *For a benzenoid B , define $\text{number_of_bay_regions}(B)$ as the number of bay regions in B*

Let $b(B)$ denote the number of bay regions in benzenoid B . Gutman & Cyvin provide the following relation on $b(B)$.

Theorem 3 (Gutman & Cyvin [5]). *Let B be a benzenoid with $e_{22}(B)$ 2-2 edges. The number of bay regions in B is given by $b(B) = e_{22}(B) - 6$.*

In SageMath we can define Invariant 16 using Theorem 3. Have the invariant-defined function $\text{number_of_bay_regions}(B)$ return $\text{number_of_2_2_edges}(B) - 6$. We want to count the number of times each type of bay region appears in a benzenoid as well.

Invariant 17 (Number of fissures). *For a benzenoid B , define $\text{fissures}(B)$ as the number of fissures in B .*

Invariant 18 (Number of bays). For a benzenoid B , define *bays* (B) as the number of bays in B .

Invariant 19 (Number of coves). For a benzenoid B , define *coves* (B) as the number of coves in B .

Invariant 20 (Number of fjords). For a benzenoid B , define *fjords* (B) as the number of fjords in B .

These structural features in a benzenoid can have an impact on its domination number. However, like finding $h(B)$, it is not immediately clear how to define a formula computing these invariants. Instead, using SageMath, we can write code to index over the list of vertices on the perimeter of a benzenoid and search for specific degree sequences. For example, when computing the number of bays in benzenoid, we filter through all vertices on the outer face and count the number of times the sequence 2, 3, 3, 2 appears. This is the specific sequence that designates a bay. The total number of times this degree sequence occurs is returned as Invariant 18. Similarly, we search for the degree sequence 2, 3, 2 to count the number of fissures. For coves: 2, 3, 3, 3, 2, and for fjords: 2, 3, 3, 3, 3, 2.

Now that the bay region invariants are defined, we focus on adding more edge invariants. The following three invariants are similar to the earlier edge invariants separated by degree of the endpoint. Now we take into account whether a particular edge lies along the perimeter or interior of a benzenoid.

Invariant 21 (Number of external 2-3 edges). For a benzenoid B , define *number_of_external_2_3_edges* (B) as the number of external 2-3 edges in B .

Invariant 22 (Number of external 3-3 edges). For a benzenoid B , define *number_of_external_3_3_edges* (B) as the number of external 3-3 edges in B .

Invariant 23 (Number of internal 3-3 edges). For a benzenoid B , define *number_of_internal_3_3_edges* (B) as the number of internal 3-3 edges in B .

Denote the above three invariants as $ex_{23}(B)$, $ex_{33}(B)$ and $in_{33}(B)$, respectively. Note that we may also define an invariant for the number of external 2-2 edges, but this already equal to an invariant we previously defined. For any benzenoid, the number of *external* 2-2 edges is equal to the *total* number of 2-2 edges, $e_{22}(B)$. A 2-2 edge may only lie on the perimeter of a benzenoid. Formulas for calculating the above invariants are given below, imported from [5]. For a benzenoid B:

- $ex_{23}(B) = 4h(B) - 2b(B) - 2n_i(B) - 4$
- $ex_{33}(B) = m(B) - in_{33}(B) - ex_{22}(B) - e_{23}(B)$
- $in_{33}(B) = h(B) + n_i(B) - 1$

The final invariants defined in this section determine the number of hexagons with varying amounts of external edges. An external edge in a hexagon is an edge that lies on the perimeter of a benzenoid. We will use these invariants to determine quantities such as the amount of hexagons in B having 5 external edges, corresponding to number of hexagons sharing only one edge with another hexagon. Another invariant will determine how many hexagons in B have 0 external edges. These hexagons are completely surrounded by other hexagons; all of their vertices are internal.

Invariant 24 (Number of hexagons with 0 external edges). *For a benzenoid B, define $faces_with_0_external_edges(B)$ as the number of hexagons in B that contain 0 external edges.*

Invariant 25 (Number of hexagons with 1 external edge). *For a benzenoid B, define $faces_with_1_external_edge(B)$ as the number of hexagons in B that contain 1 external edge.*

Invariant 26 (Number of hexagons with 2 external edges). *For a benzenoid B, define $faces_with_2_external_edges(B)$ as the number of hexagons in B that contain 2 external edges.*

Invariant 27 (Number of hexagons with 3 external edges). For a benzenoid B , define $faces_with_3_external_edges(B)$ as the number of hexagons in B that contain 3 external edges.

Invariant 28 (Number of hexagons with 4 external edges). For a benzenoid B , define $faces_with_4_external_edges(B)$ as the number of hexagons in B that contain 4 external edges.

Invariant 29 (Number of hexagons with 5 external edges). For a benzenoid B , define $faces_with_5_external_edges(B)$ as the number of hexagons in B that contain 5 external edges.

Determining these invariants involves initializing a counter and finding the endpoints appearing on the perimeter of a benzenoid. The counter is updated dependent on the specific invariant we are seeking and the degree sequence. This is just the idea behind the code. A defined function for each invariant is provided in Appendix A.

After the invariants discussed in this section are defined in SageMath, CONJECTURING has the majority of ingredients needed to begin outputting conjectures that provide upper bounds on $\gamma(B)$. Generating the conjectures is half of the battle. Our end goal is to prove interesting conjectures, or provide a counterexample to refine a bound. In the next section we discuss the techniques used to prove most generated conjectures.

1.4 Methods of proof

There are two useful methods unique to benzenoids, as a class of graphs, that are used when proving CONJECTURINGS's conjectures. The first is the removable hexagon lemma, which is stated as Lemma 1 in the previous section. The second technique is to tighten the domination number bound on classes of benzenoids. In particular, one class of interest for generating upper bounds are *catacondensed* benzenoids. Catacondensed benzenoids have the property of every carbon ring in the hydrocarbon molecule connects to

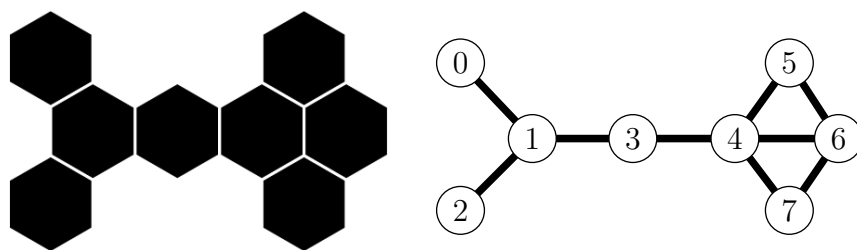


Figure 1.8: Example of a benzenoid I and its inner dual I_{id} .

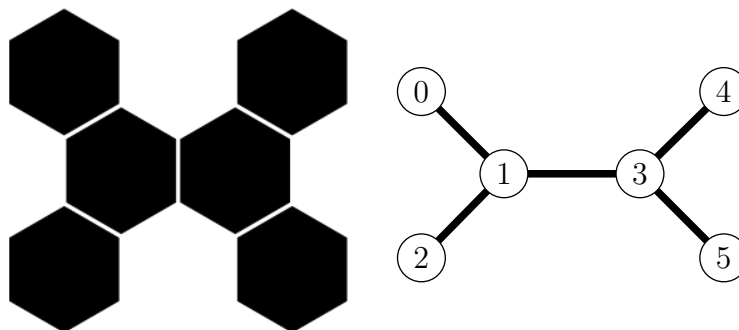


Figure 1.9: Another benzenoid J and its inner dual J_{id} .

another ring via one carbon bond. In the plane, a catacondensed benzenoid is embedded with all hexagons in the benzenoid's cluster sharing a single edge. The graph theoretic interpretation of the catacondensed property is given as the following definition.

Definition 10. *A benzenoid B is catacondensed if its corresponding inner dual is a tree.*

To fully explain the catacondensed property it is necessary to investigate a benzenoid's inner dual, given in Definition 9, and determine if the inner dual is a *tree*. To illustrate the idea of the inner dual the following two figures provide a benzenoid with its corresponding inner dual. The benzenoid in Figure 1.8 is not catacondensed and benzenoid in Figure 1.9 is catacondensed. These visuals will help explain the definition of a tree.

Definition 11. *A tree is an acyclic, connected graph.*

Notice in Figure 1.8 that the inner dual I_{id} contains four vertices, the furthest to

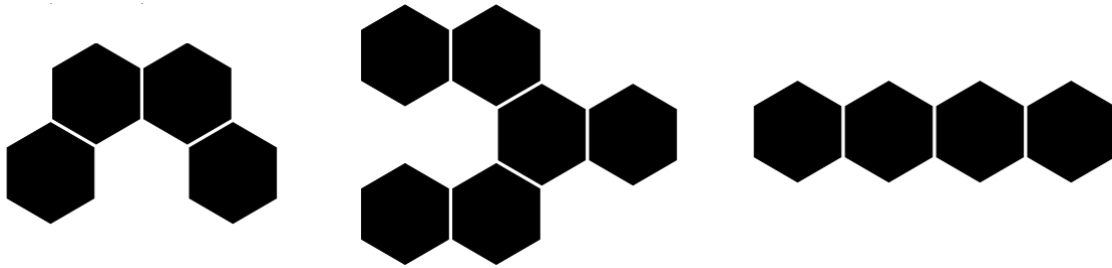


Figure 1.10: Additional catacondensed benzenoids

the right of the margin, which are connected in a closed loop. These closed loops are cycles. If we select any two vertices on this cycle we will find two independent paths connecting the two chosen vertices. An *acyclic* graph does not contain this property. A *connected* graph is a graph in which every vertex is incident to at least one edge.

Between benzenoids I and J in Figures 1.8 and 1.9 there is a significant difference in their inner duals. We have observed that the inner dual of I contains a cycle, and therefore is not a tree. Let's turn our attention to the inner dual of J. Every vertex in J_{id} is connected as each vertex is incident to at least one edge. Selecting an arbitrary vertex, we see that this vertex is not part of a closed chain. There is not a way to find two independent paths between any two vertices. So J_{id} is acyclic. As J_{id} is acyclic and connected, J_{id} is a tree. Further, as J's corresponding inner dual is a tree, we conclude that J is catacondensed.

Additional examples of catacondensed benzenoids are shown in Figure 1.10. It is straightforward to check that their inner duals are trees.

Catacondensed benzenoids allow us to define two new invariants based on the structure of the inner dual. The first invariant counts the number of *leaves* a inner dual, while the second counts the number of *branching hexagons* an embedding.

Definition 12. A hexagon H in benzneoid B is called a leaf if the vertex corresponding to H in the inner dual B_{id} has a degree of 1.

By Corollary 2.12 in [2], every nontrivial tree contains two leaves. By extension, every catacondensed benzenoid contains two leaves.

Definition 13. A hexagon H in B is called a branching hexagon if the vertex corresponding to H in the inner dual B_{id} has a degree of 3 or a degree of 2 where the two shared edges in H do not form a pair.

In the definition above we mention *shared edges* forming a *pair*. We called an edge e in hexagon H of benzenoid B *shared* if e also belongs to another, different hexagon H' in B . For an edge existing in only one hexagon we say the edge is independent. Pairs of edges in a benzenoid are specific to a single hexagon. For a hexagon H in the embedding of B , we call the parallel edges in H pairs. Pairs of edges are antipodal, meaning they are on opposite sides of H . This covers the terminology used in the following invariants.

Invariant 30 (Number of leaves). For a catacondend benzenoid B , define $leaves(B)$ as the number of leaves in B .

Invariant 31 (Branching number). For a catacondensed benzenoid B , define $branching_number(B)$ as the number of branching hexagons of B .

The two invariants above are denoted $h_l(B)$ and $h_b(B)$, respectively. It is helpful to program a function returning the inner dual of a benzenoid to help code Invariant 30. First return B 's inner dual B_{id} . Count the number of degree-2 vertices in B_{id} and return that integer as $h_l(B)$. Similarly, for Invariant 31, the number of vertices of degree-3 is determined in B_{id} . This amount is added to the number of degree-2 vertices in B_{id} whose edges in the corresponding hexagon of B do not form a pair. To find the number of times this occurs we search the embedding of B looking for the degree sequence 2, 3, 3, 2. This sequence corresponds to a branching path in the inner dual and resembles a bay in the embedding of B . The total quantity is returned as $h_b(B)$.

Catacondensed benzenoids are a superset to an introductory class of benzenoids known as *linear benzenoids*. Linear refers to the hexagons in an embedding maintaining parallel lines among pairs of edges. A linear benzenoid is a catacondensed benzenoid with 0 branching hexagons. The linear benzenoid with the smallest number of

hexagons is benzene. Linear benzenoids are denoted L_n where n identifies the number of hexagons. Figure 1.11 shows L_2, L_3 , and L_4 . Computing $\gamma(L_2), \gamma(L_3)$, and $\gamma(L_4)$ reveals a pattern relating the domination number of a linear benzenoid with its hexagon count. The relation is proven in the following statement.

Proposition 1. *For a linear benzenoid B with $h(B)$ hexagons, $\gamma(B) = h(B) + 1$.*

Proof. The proposition is true for small linear chains. When $h(B) = 1$, benzenoid B is benzene and $\gamma(B) = 2 = h(B) + 1$. Notice that when $h(B) = 2$, benzenoid B is anthracene and $\gamma(B) = 3 = h(B) + 1$. Assume the proposition is true for benzenoid B with $h(B) = k - 1 \geq 2$ hexagons. Let B be a benzenoid with $h(B) = k$ hexagons. Suppose hexagons H_1 and H_2 in B are at maximum distance with respect to B_{id} .

Without loss of generality, create an antipodal dominating set $D(B)$ by first choosing vertex v in H_1 that has maximum eccentricity in B . By construction of linear benzenoids, vertex v is antipodal to degree-3 vertex w . Choose w as the next vertex in $D(B)$ and repeatedly select antipodal vertices as the remaining elements of $D(B)$ ending with vertex z in H_2 that has maximum eccentricity in B . Then $D(B)$ is a minimum dominating set of B . This construction dominates all vertices in B without overlap.

As $k - 1 \geq 2$, and B is a linear benzenoid, there exists at least one hexagon separating H_1 and H_2 . Let H_3 be one hexagon between H_1 and H_2 . Removing H_3 from B disconnects B into two components. Let L be the component in $B - H_3$ containing H_1 and R be the component in $B - H_3$ containing H_2 . We can form minimum dominating sets for L and R by setting $D(L) := V(L) \cap D(B)$ and $D(R) := V(R) \cap D(B)$. Then $D(B) = D(L) \cup D(R)$.

By the inductive hypothesis $\gamma(L) = h(L) + 1 = |V(L) \cap D(B)|$ and $\gamma(R) = h(R) + 1 =$



Figure 1.11: Linear benzenoids $L_2, L_3,$ and L_4 : naphthalene, anthracene, and naphthacene

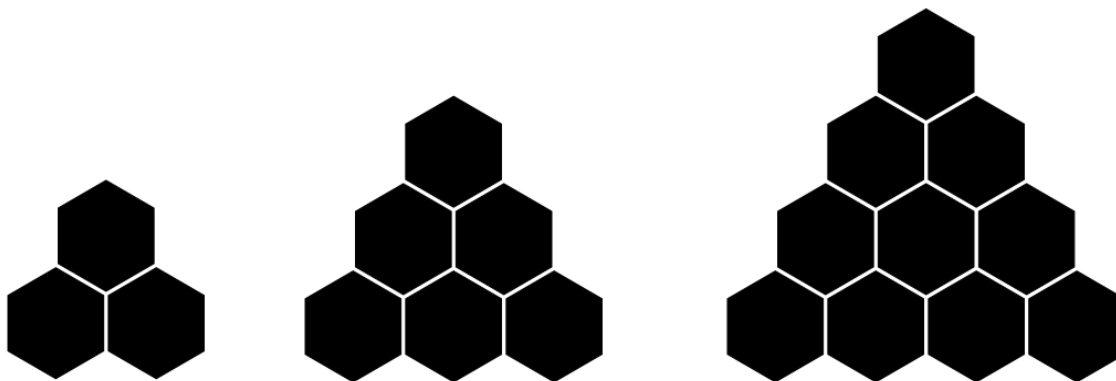


Figure 1.12: From left: Triangulenes T_2, T_3 and T_4

$|V(R) \cap D(B)|$. Notice that

$$\begin{aligned}
 |D(B)| &= |V(L) \cap D(B)| + |V(R) \cap D(B)| \\
 &= h(L) + 1 + h(R) + 1 \\
 &= (h(L) + h(R) + 1) + 1 \\
 &= h(B) + 1.
 \end{aligned}$$

As $D(B)$ is a minimum dominating set, $|D(B)| = \gamma(B)$. It follows by mathematical induction that $\gamma(B) = h(B) + 1$. \square

Outside of catacondensed benzenoids, another class of benzenoids are *triangulenes*. A triangulene is a pyramidal benzenoid of i levels. Each level is a row in the graph embedding. Benzene is triangulene of smallest order and has 1 level. An arbitrary triangulene is denoted T_i and named triangulene- i . The second and third level triangulenes are T_2 and T_3 . Figure 1.12 shows T_2 and T_3 along with triangulene-4.

The last class of benzenoids we discuss are *circulenes*. A circulene is a benzenoid

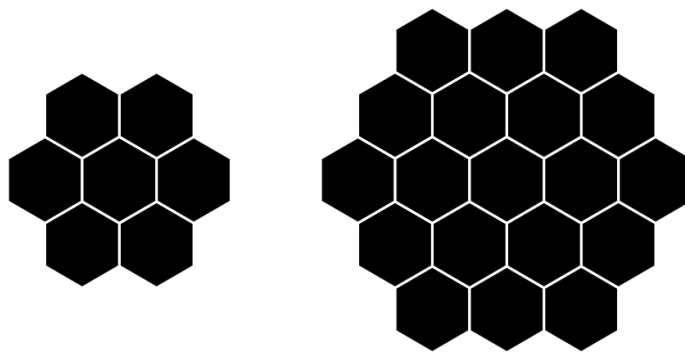


Figure 1.13: Circulenes C'_2 (left) and C'_3 (right)

that can be constructed from benzene by adding some number of layers in an iterative method. Each additional layer in a circulene adds a hexagon to each edge on the perimeter of the previous circulene. The circulene with the smallest order is benzene. A circulene with n layers is denoted as C'_n and named circulene- n . Recall that C_n , without the prime, denotes a cycle graph. Circulene C'_2 is coronene and C'_3 is circulene-3. Figure 1.13 shows C'_2 and C'_3 .

In the next section we will prove invariant conjectures given by CONJECTURING. Some conjectures are proven for any arbitrary benzenoids and others are tighter bounds on classes of benzenoids.

1.5 Results

After all invariants are defined in SageMath, and a few initial benzenoids are programmed, we are ready to generate conjectures. Load CONJECTURING into SageMath to run the conjecture-generating `conjectures()` function. The process of adding benzenoids and functions in SageMath for use by CONJECTURING is detailed in Section 2.2. CONJECTURING uses these known benzenoids and invariants as ingredients for outputting automated conjectures. Specifically, the program returns conjectures that bounds any invariant we specify as the *invariant of interest*. For this study, all of the conjectures are generated with the domination number as the invariant of interest. The domination

number is defined in Section 1.3 as Invariant 3. We choose to generate upper bounds on $\gamma(B)$ as opposed to lower bounds. As we discussed earlier, the domination number has a natural lower bound of $\frac{n(B)}{4}$.

We will prove two bounds on $\gamma(B)$ found from observation before presenting automated conjectures from CONJECTURING. The first bound makes use of the removable hexagon lemma to bound the domination number in an iterative way. The domination number of benzenoid B' obtained by subtracting a removable hexagon H from benzenoid B will have a resulting smaller domination number by at most 2. That is $\gamma(B') \geq \gamma(B) - 2$. Adding 2 to both sides of the inequality results in the following upper bound.

Proposition 2. *Let B be a non-trivial benzenoid and let B' be a benzenoid obtained by eliminating a removable hexagon from B . Then $\gamma(B) \leq \gamma(B') + 2$.*

Proof. By Lemma 1 every non-trivial benzenoid has a removable hexagon. Let B' be the benzenoid obtained from eliminating removable hexagon H from B . Assume the maximal case in which B' is obtained by removing four vertices from B . If the upper bound holds for the maximal case it must hold for all cases. Attach H to B' to form B . Select a vertex in B not in B' that is adjacent to two vertices also not in B' . By construction there are two such vertices. One of these two vertices must be in a minimal dominating set. Then there is one remaining vertex that is not dominated. Add this vertex to the dominating set, and it must be that $\gamma(B) \leq \gamma(B') + 2$. \square

The next upper bound on $\gamma(B)$ is observed from choosing two *antipodal* vertices from each hexagon in B to be in a dominating set of B . Antipodal vertices are opposite in each hexagon. These vertices do not share any common neighbors in the hexagons they dominate.

Proposition 3 (Upper bound for hexagons). *Let B be a benzenoid with $h(B)$ hexagons. Then $\gamma(B) \leq 2h(B)$.*

Proof. Notice that each hexagon can be dominated by 2 antipodal vertices. Let $D(B)$ be any set of vertices in $V(B)$ which includes 2 antipodal vertices from each hexagon. Then $D(B)$ is a dominating set. It must be that $\gamma(B) \leq |D(B)| \leq 2h(B)$. \square

A useful way to generate stronger bounds on classes of benzenoids is to restrict the benzenoids known to CONJECTURING. Building a list of benzenoids that reside in one class will limit the knowledge of CONJECTURING so the program will only generate conjectures for that class of benzenoids. First we generate and prove conjectures for all benzenoids. Then we compute a separate conjecture run with only catacondensed benzenoids defined. In this way CONJECTURING will return conjectures specifically applicable to catacondensed benzenoids.

CONJECTURING outputs automated conjectures in terms of the function names defined for each invariant. Our results follow a conjecture-proposition-proof format. First the generated conjecture is shown, followed by a proposition relating the function names to standard notation. Then a proof is given after each proposition.

1.5.1 General benzenoids

Conjecture 1.

$$\text{domination_number}(B) \leq B.\text{size}()/2$$

Proposition 4. *Let B be a benzenoid with $h(B)$ hexagons and $m(B)$ edges. Then $\gamma(B) \leq \frac{m(B)}{2}$.*

Proof. We will prove this using mathematical induction. For the base case when $h(B) = 1$ we note $m(B) = 6$ and $\gamma(B) = 2 \leq \frac{6}{2} = 3$. So the base case holds. Assume the statement holds for a benzenoid with $k - 1 \geq 1$ hexagons. We will show it holds for a benzenoid with k hexagons. Let B be a benzenoid with $h(B) = k$ hexagons. By Lemma 1, B has a removable hexagon H . Let $B' = B - H$ be the benzenoid formed from removing H in B . Then $h(B') = h(B) - 1 = k - 1$ and we may apply the inductive hypothesis. We have multiple cases. First suppose that B' is formed by removing one edge in $E(B)$. Then we

find

$$\gamma(B) \leq \gamma(B') \leq \frac{m(B')}{2} = \frac{m(B) - 1}{2} \leq \frac{m(B)}{2},$$

which is what we needed to show. Now suppose the case that B' is formed by removing two edges in $E(B)$. Notice that

$$\begin{aligned} \gamma(B) &\leq \gamma(B') + 1 \\ &\leq \frac{m(B')}{2} + 1 \\ &\leq \frac{m(B) - 2}{2} + 1 = \frac{m(B)}{2}. \end{aligned}$$

Following a similar set of steps, when B' is formed by removing three edges in $E(B)$ we have

$$\begin{aligned} \gamma(B) &\leq \gamma(B') + 1 \\ &\leq \frac{m(B')}{2} + 1 \\ &= \frac{m(B) - 3}{2} + 1 \leq \frac{m(B)}{2}. \end{aligned}$$

When $m(B') = m(B) - 4$, the induction works out as well:

$$\begin{aligned} \gamma(B) &\leq \gamma(B') + 1 \\ &\leq \frac{m(B')}{2} + 1 \\ &= \frac{m(B) - 4}{2} + 1 \leq \frac{m(B)}{2}. \end{aligned}$$

Finally, consider when B' is constructed by removing a hexagon by removing five edges

from $E(B)$. It is clear that

$$\begin{aligned}\gamma(B') &\leq \gamma(B) + 2 \\ &\leq \frac{m(B)}{2} + 2 \\ &= \frac{m(B') - 5}{2} + 2 \leq \frac{m(B')}{2}.\end{aligned}$$

It follows by induction that $\gamma(B) \leq \frac{m(B)}{2}$. □

Conjecture 2.

$$\textit{domination_number}(B) \leq \textit{smaller_bipartite_set}(B) - 1$$

Proposition 5. *For any benzenoid B , $\gamma(B) \leq |\mathcal{B}_s(B)| - 1$.*

Proof. We will prove this by induction on the number of hexagons in B . Notice that when $h(B) = 1$, B is benzene. It is clear that $\gamma(B) = 2 \leq 3 - 1 = |\mathcal{B}_s(B)| - 1$. Similarly, when $h(B) = 2$, B is anthracene and we see that $\gamma(B) = 3 \leq 10 - 1 = |\mathcal{B}_s(B)| - 1$. Assume that the statement is true for a benzenoid with $k - 1$ hexagons where $k - 1 \geq 2$. We will show the statment holds for k hexagons. Let B be a benzenoid with k hexagons. By Lemma 1, there is a removable hexagon H in B . Let $B' = B - H$ be the benzenoid defined by removing H from B . We have multiple cases. First, suppose that H has two degree-2 vertices. Since H has only two degree-2 vertices, they must be adjacent. As the vertices are adjacent, each one belongs to a separate bipartite set. Notice that

$$|\mathcal{B}_s(B')| = |\mathcal{B}_s(B)| - 1 \text{ and } |\mathcal{B}_l(B')| = |\mathcal{B}_l(B)| - 1.$$

By the inductive hypothesis, $\gamma(B') \leq |\mathcal{B}_s(B')| - 1$. It must be that that

$$\begin{aligned} \gamma(B) &\leq \gamma(B') + 1 \\ &\leq |\mathcal{B}_s(B')| - 1 + 1 \\ &= |\mathcal{B}_s(B)| - 1. \end{aligned}$$

Now suppose the case that H has four degree-2 vertices. This case is very similar to the first one. It must be that removing H removes two vertices from each bipartite set. Then

$$|\mathcal{B}_s(B')| = |\mathcal{B}_s(B)| - 2 \text{ and } |\mathcal{B}_l(B')| = |\mathcal{B}_l(B)| - 2.$$

Again, by the inductive hypothesis $\gamma(B') \leq |\mathcal{B}_s(B')| - 1$. Observe that

$$\begin{aligned} \gamma(B) &\leq \gamma(B') + 1 \\ &\leq |\mathcal{B}_s(B')| - 1 + 2 \\ &= |\mathcal{B}_s(B)| - 1. \end{aligned}$$

Consider the final case where H has three degree-2 vertices. Subcase 1: Only one of the three vertices that we remove from B are in $\mathcal{B}_s(B)$. Then $|\mathcal{B}_s(B')| = |\mathcal{B}_s(B)| - 1$ and

$$\begin{aligned} \gamma(B) &\leq \gamma(B') + 1 \\ &\leq (|\mathcal{B}_s(B')| - 1) + 1 \\ &= |\mathcal{B}_s(B)| - 1. \end{aligned}$$

Subcase 2: Two of the three vertices we remove from B are in $\mathcal{B}_s(B)$. Notice that

$|\mathcal{B}_s(B')| = |\mathcal{B}_s(B)| - 1$ and

$$\begin{aligned}
 \gamma(B) &\leq \gamma(B') + 1 \\
 &\leq (|\mathcal{B}_s(B')| - 1) + 1 \\
 &= |\mathcal{B}_s(B)| - 2 \\
 &\leq |\mathcal{B}_s(B)| - 1.
 \end{aligned}$$

It follows by induction that $\gamma(B) \leq |\mathcal{B}_s(B)| - 1$. □

Lemma 2. For all benzenoids B , $|\mathcal{B}_s(B)| \leq \frac{n(B)}{2}$.

Proof. By construction, benzenoids are bipartite. The vertices in $V(B)$ may be split into two bipartite sets $\mathcal{B}_s(B)$ and $\mathcal{B}_l(B)$ bounded by the inequality $|\mathcal{B}_s(B)| \leq |\mathcal{B}_l(B)|$. At most, $\mathcal{B}_s(B)$ will contain half of the elements of $V(B)$. Then the cardinality of $\mathcal{B}_s(B)$ is bounded by half the order. □

Corollary 2. Proposition 5 implies that, for a benzenoid B ,

$$\gamma(B) \leq \frac{n(B)}{2} - 1.$$

Proof. By Lemma 2,

$$|\mathcal{B}_s(B)| \leq \frac{n(B)}{2}.$$

Then by Proposition 5,

$$\gamma(B) \leq |\mathcal{B}_s(B)| - 1 \leq \frac{n(B)}{2} - 1.$$

□

1.5.2 Catacondensed benzenoids

Conjecture 3.

$$\text{domination_number}(B) \leq B.\text{size}()/2 - \text{hexagons}(B)$$

Proposition 6. For any catacondensed benzenoid B ,

$$\gamma(B) \leq \frac{m(B)}{2} - h(B).$$

Remark. Proposition 6 is equivalent to $\gamma(B) \leq \frac{3h(B)+1}{2}$.

Observe. In [5], the formula for the number of edges is given by $m(B) = 5h(B) + 1 - n_i(B)$. As B is a catacondensed benzenoid, there are 0 internal vertices. Then Conjecture 6 may be expressed as $\gamma(B) \leq \frac{5h(B)+1}{2} - h(B)$. Notice that

$$\gamma(B) \leq \frac{5h(B) + 1}{2} - h(B) = \frac{5h(B) + 1}{2} - \frac{2h(B)}{2} = \frac{3h(B) + 1}{2}.$$

□

We now write the equivalency as the following proposition.

Proposition 7. For any catacondensed benzenoid B ,

$$\gamma(B) \leq \frac{3h(B) + 1}{2}.$$

Proof. We have two cases. First assume that there are no branching hexagons. If this is the case, we may divide our benzenoid into two parts in which one has an even number of hexagons. By induction, assume this is true for benzenoids with less than $h(B)$ hexagons. Let B be a catacondensed benzenoid with $h(B)$ hexagons. Divide B into two parts, B_1 and B_2 where, without loss of generality, B_2 has an even number of

hexagons. By the inductive hypothesis, we have that

$$\gamma(B_1) \leq \frac{3h(B_1) + 1}{2}$$

and

$$\gamma(B_2) \leq \frac{3h(B_2) + 1}{2} \leq \frac{3h(B_2)}{2}.$$

Then it must be that

$$\begin{aligned} \gamma(B) &\leq \gamma(B_1) + \gamma(B_2) \\ &\leq \frac{3h(B_1) + 1}{2} + \frac{3h(B_2)}{2} \\ &= \frac{3(h(B_1) + h(B_2)) + 1}{2} \\ &= \frac{3h(B) + 1}{2}. \end{aligned}$$

Now suppose the case in which we have a branching hexagon. Divide the benzenoid into three pieces, B_1 , B_2 , and B_3 , at the midpoint of the branching hexagon. By the inductive hypothesis,

$$\gamma(B_i) \leq \frac{3h(B_i) + 1}{2}.$$

Observe that

$$\begin{aligned} \gamma(B) &\leq \gamma(B_1) + \gamma(B_2) + \gamma(B_3) \\ &\leq \frac{3h(B_1) + 1}{2} + \frac{3h(B_2) + 1}{2} + \frac{3h(B_3) + 1}{2} \\ &= \frac{3(h(B) - 1) + 3}{2} = \frac{3h(B)}{2} \\ &\leq \frac{3h(B) + 1}{2}, \end{aligned}$$

which is what we needed to show. □

Conjecture 4.

$$\text{domination_number}(B) \leq \text{number_of_2_2_edges}(B)/2 + \text{hexagons}(B) - 2$$

Proposition 8. For a catacondensed benzenoid B ,

$$\gamma(B) \leq \frac{e_{22}(B)}{2} + h(B) - 2.$$

Proof. We have two cases. First assume that there is a branching hexagon. Then we may split the benzenoid into three parts and find that $\gamma(B) = \gamma(B_1) + \gamma(B_2) + \gamma(B_3)$. We have the equality since splitting the benzenoid at the branching hexagon leaves zero remaining vertices undominated. Also, the number of external 2-2 edges of B is given by $e_{22}(B) = e_{22}(B_1) + e_{22}(B_2) + e_{22}(B_3)$ and the number of hexagons in B is given by $h(B) = h(B_1) + h(B_2) + h(B_3) + 1$. Then

$$\begin{aligned} \gamma(B) &= \gamma(B_1) + \gamma(B_2) + \gamma(B_3) \\ &\leq \left(\frac{e_{22}(B_1)}{2} + h(B_1) - 2 \right) + \left(\frac{e_{22}(B_2)}{2} + h(B_2) - 2 \right) + \left(\frac{e_{22}(B_3)}{2} + h(B_3) - 2 \right) \\ &\leq \frac{e_{22}(B) + 9}{2} + (h(B) - 1) - 6 \\ &= \frac{e_{22}(B)}{2} + h(B) - 2 - \frac{1}{2} \\ &\leq \frac{e_{22}(B)}{2} + h(B) - 2, \end{aligned}$$

completing the first part of the proof. Now suppose that there is not a branching hexagon in B . Consider splitting the benzenoid at a hexagon beside a hexagon on the outer face

of B . Then

$$\begin{aligned}
\gamma(B) &= \gamma(B_1) + \gamma(B_2) \\
&= \left(\frac{e_{22}(B_1)}{2} + h(B_1) - 2 \right) + \left(\frac{e_{22}(B_2)}{2} + h(B_2) - 2 \right) \\
&\leq \frac{e_{22}(B) + 6}{2} + (h(B) - 1 - 4) \\
&= \frac{e_{22}(B)}{2} + h(B) - 2.
\end{aligned}$$

□

For the next proposition it will be helpful to show the following lemma which states every catacondensed benzenoid has a minimum dominating set containing one vertex from the independent edge in leaf H opposite the shared edge of H .

Remark. For edge e in benzenoid B , we denote the edge that forms a pair with e in hexagon H by e'_H .

Lemma 3. Let H be a leaf hexagon in catacondensed benzenoid B , and let e be its shared edge. Then B has a minimum dominating set S that contains one vertex from e'_H .

Proof. Let H be a leaf hexagon in a benzenoid B . Let $\{v_1, \dots, v_6\}$ be the vertices of the hexagon, and let $\{(v_i, v_{i+1}) : i \in [6]\}$ (addition done modulo 6, so $i + 1 = 1$ when $i = 6$) be the set of edges in H . Without loss of generality, let $e = \{v_1, v_2\}$ be the shared edge of H , and let $e'_H = \{v_4, v_5\}$. We will construct a minimum dominating set that contains either v_4 or v_5 . Let S be a minimum dominating set that contains neither. This implies that $\{v_3, v_6\} \in S$. We now split into two cases.

Case 1: Suppose $\{v_1, v_2\} \cap S = \emptyset$. In this case, $S' = S \setminus \{v_3, v_6\} \cup \{v_1, v_4\}$ is a minimum dominating set containing v_4 .

Case 2: Suppose $\{v_1, v_2\} \cap S \neq \emptyset$. Without loss of generality, suppose $v_1 \in S$. Then $S' = S \setminus \{v_6\} \cup \{v_5\}$ is a minimum dominating set containing v_4 . □

Conjecture 5.

$$\text{domination_number}(B) \leq \text{hexagons}(B) + \text{branching_number}(B) + 1$$

Proposition 9. For a catacondensed benzenoid B ,

$$\gamma(B) \leq h(B) + h_b(B) + 1,$$

where $h_b(B)$ denotes the branching number of B .

Proof. We will prove this by induction on the number of hexagons in B . When $h(B) = 1$, we have that

$$\gamma(B) = 2 \leq 2 = 1 + 0 + 1 = h(B) + h_b(B) + 1.$$

Similarly, when $h(B) = 2$ we find

$$\gamma(B) = 3 \leq 4 = 2 + 1 + 1 = h(B) + h_b(B) + 1.$$

Assume the statement holds for a benzenoid with $k - 1 \geq 2$ hexagons. Let B be a catacondensed benzenoid with $h(B) = k$ hexagons. By Lemma 3, we know that there is a leaf, say v_1 , in the inner dual B_{id} whose neighbor, say v_2 , has a degree-2 or a "twin leaf", say v_3 , i.e. a leaf adjacent to v_2 . We can handle the two cases separately.

Case 1. First suppose that v_2 has a degree of 2. Let B' be the benzenoid obtained from B by removing the leaf hexagon H_1 , i.e. removing all four vertices in H_1 not belonging to the shared edge of H_1 . Let $h(B')$ and $h_b(B')$ denote the number of hexagons and branching hexagons in B' respectively. Note that $h(B') = h(B) - 1$ and $h_b(B') = h_b(B)$. Suppose first that H_2 is not a branching hexagon in B . By definition, the two shared edges of H_2 form a pair. Let e be the edge H_2 shares with H_3 ; then, e'_{H_2} is the edge H_2 shares with H_1 in B . Using Lemma 3, there is a minimum dominating set that contains one vertex from e'_{H_2} , which implies that $\gamma(B) \leq \gamma(B') + 1$. Using the induction

hypothesis on B' , this gives $\gamma(B) \leq h(B') + h_b(B') + 1 + 1 = h(B) + h_b(B) + 1$. Now suppose that H_2 is a branching hexagon in B . Then H_2 is not a branching hexagon in B' , as it has a degree of 1 in B'_{id} . Consequently, $h(B') = h(B) - 1$ and $h_b(B') = h_b(B) - 1$. Furthermore, $\gamma(B) \leq \gamma(B') + 2$. Using the induction hypothesis on B' yields $\gamma(B) \leq h(B') + h_b(B') + 1 + 2 = h(B) + h_b(B) + 1$, as required.

Case 2. Now suppose that v_2 has degree-3 and v_1 and v_3 are twin leaves adjacent to v_2 . Suppose v_4 is the third vertex adjacent to v_2 , and we may assume that v_4 is not a leaf. (If it is, the benzenoid B has $h(B) = 4$, $h_b(B) = 1$ and $\gamma(B) = 6$, thus proving the bound.) Let $\{x_1, x_2, \dots, x_6\}$ be the vertices in H_2 . Without loss of generality, let the three shared edges be $e_1 = \{x_1, x_2\}$, $e_2 = \{x_3, x_4\}$, and $e_3 = \{x_5, x_6\}$. Let e_1 be an edge in H_1 and e_3 be an edge in H_3 . We will consider the benzenoid B' obtained by removing H_1 and H_3 , i.e. removing all vertices belonging to either H_1 or H_3 except the endpoints of e_1 and e_3 . Note that v_2 is now a leaf in inner dual B'_{id} , and by using Lemma 3 there is a minimum dominating set in B' that contains either x_1 or x_6 (the endpoints of the edge that forms a pair with e_2). This implies $\gamma(B) \leq \gamma(B') + 3$. Now, we have $h(B') = h(B) - 2$ and $h_b(B') = h_b(B) - 1$, so by the induction hypothesis for B' we see that $\gamma(B) \leq h(B') + h_b(B') + 4 = h(B) + h_b(B) + 1$, which is what we needed to show. \square

Definition 14. Denote $h_b(B)_i$, with $i \in \{2, 3\}$, as the number of branching hexagons that have exactly i shared edges.

Note that $h_b(B) = h_b(B)_2 + h_b(B)_3$. This refinement to the number of branching hexagons will help us prove the following corollary.

Corollary 3. Proposition 9 implies that

$$\gamma(B) \leq h(B) + ex_{33}(B) + 1,$$

where $ex_{33}(B)$ denotes the number of external edges connected by two degree-3 vertices.

Proof. Using Proposition 9, it is sufficient to show that $ex_{33}(B) \geq h_b(B)$. Note that no shared edges are counted by $ex_{33}(B)$ since shared edges must be internal edges. If a hexagon in B is either a leaf hexagon or a non-branching hexagon with exactly 2 shared edges, then it does not contain any edges that are counted by $ex_{33}(B)$. On the other hand, a branching hexagon containing 2 shared edges contributes exactly 1 to $ex_{33}(B)$, which a branching hexagon containing the 3 shared edges contributes exactly 3. Then $ex_{33}(B) \geq h_b(B)_2 + 3h_b(B)_3 \geq h_b(B)$. By Proposition 9, $\gamma(B) \leq h(B) + h_b(B) + 1 \leq h(B) + ex_{33}(B) + 1$. \square

Conjecture 6.

$$domination_number(B) \leq B.order()/3$$

Proposition 10. *For a catacondensed benzenoid B ,*

$$\gamma(B) \leq \frac{n(B)}{3}.$$

Proof. We will prove this statement using mathematical induction on $h(B)$. The statement can be verified easily for $h(B) \leq 3$, so suppose $h(B) \geq 4$. Let B be a catacondensed benzenoid with $h(B)$ hexagons and inner dual B_{id} . As B is catacondensed, B_{id} is either a path or contains a vertex of degree-3. We handle the two cases separately.

Case 1. Let B_{id} be a path, say $B_{id} = (v_1, v_2, \dots, v_n)$ and let B_3 be the subgraph of B induced by the hexagons H_1, H_2 and H_3 . Let $e \in B_3$ be the (unique) shared edge of H_3 that also belongs to H_4 . Let B' be the benzenoid obtained from B by deleting every vertex in B_3 except the two endpoints of e . Using the inductive hypothesis, $\gamma(B') \leq \frac{n(B)-12}{3}$. We know that $\gamma(B_3) = 4$, which implies $\gamma(B) \leq \frac{n(B)-12}{3} + 4 = \frac{n(B)}{3}$, as required.

Case 2. Suppose B_{id} contains a degree-3 vertex v , and consider the hexagon H in B corresponding to v . If e and f are any two shared edges in H , then e and f do not share a common vertex, as it would result in a triangle in the inner dual B_{id} . Let e_1, e_2 and e_3 be the three independent edges in H , and let f_1, f_2 and f_3 be the three shared edges

in H . Let G be the graph obtained by removing edges e_1 , e_2 and e_3 from B . Then G is a disjoint union of three benzenoids, say B_1 , B_2 and B_3 (containing edges f_1 , f_2 and f_3 respectively), each of which contain a strictly smaller number of hexagons than B . Moreover, $\gamma(B) \leq \gamma(B_1) + \gamma(B_2) + \gamma(B_3)$. By the inductive hypothesis, $\gamma(B_i) \leq \frac{n(B_i)}{3}$ for $i \in \{1, 2, 3\}$. This implies $\gamma(B) \leq \frac{n(B_1)}{3} + \frac{n(B_2)}{3} + \frac{n(B_3)}{3} = \frac{n(B)}{3}$, as required. \square

1.6 More conjectures

Finally, we compile a list of upper bound conjectures on the domination number for a benzenoid that are unproven. These conjectures are separated into two tables dependent on if they apply to general benzenoids or catacondensed benzenoids, similar to the result sections above. One conjecture in particular is of interest: Conjecture 6 in Table 1.1. We showed in Proposition 6 that the bound $\gamma(B) \leq \frac{m(B)}{2} - h(B)$ holds for catacondensed benzenoids. Proving the bound holds for all benzenoids is a \$10 USD prize problem to be awarded by C. E. Larson¹. As the conjecture is of significance it is stated independently below.

Proposition (Prize problem). *For any benzenoid B ,*

$$\gamma(B) \leq \frac{m(B)}{2} - h(B).$$

¹See Abstract for contact information.

1.	<code>domination_number(B)</code>	\leq	<code>hexagons(B) + number_of_external_3_3_edges(B) - 1</code>
2.	<code>domination_number(B)</code>	\leq	<code>1/2*number_of_2_2_edges(B) + number_of_internal_3_3_edges(B) - 1</code>
3.	<code>domination_number(B)</code>	\leq	<code>-number_of_external_3_3_edges(B) + 1/2*B.size()</code>
4.	<code>domination_number(B)</code>	\leq	<code>1/2*degree_2_vertices(B) + 2*number_of_2_3_edges(B)</code>
5.	<code>domination_number(B)</code>	\leq	<code>maximum(hexagons(B), number_of_2_3_edges(B)) + 1</code>
6.	<code>domination_number(B)</code>	\leq	<code>B.size()/2 - hexagons(B)</code>

Table 1.1: Open conjectures for any benzenoid

1.	<code>domination_number(B)</code>	\leq	<code>number_of_2_3_edges(B)+2</code>
2.	<code>domination_number(B)</code>	\leq	<code>B.size()/coves(B)</code>
3.	<code>domination_number(B)</code>	\leq	<code>maximum(degree_3_vertices(B), sqrt(order(B)))</code>
4.	<code>domination_number(B)</code>	\leq	<code>maximum(degree_3_vertices(B), hexagons(B) + 1)</code>
5.	<code>domination_number(B)</code>	\leq	<code>bays(B) + 1/2*degree_2_vertices(B)</code>
6.	<code>domination_number(B)</code>	\leq	<code>maximum(hexagons(B), number_of_3_3_edges(B)) + 1</code>

Table 1.2: Open catacondensed conjectures

Chapter 2

Property probe

Now we generate property based conjectures on benzenoids. The method of defining functions in SageMath changes from invariant-defined functions to property-defined functions. Property-defined functions differ from invariants in that property functions return a truth value of True or False as opposed to a numeric value. A property based conjecture is proposed as an 'if P, then Q' statement: If a benzenoid has property P, then the benzenoid has property Q. A property conjecture is not limited to two properties. Generated conjectures often are combinations of multiple properties and logical operators such as *and*, *or*, and *xor*. The CONJECTURING program generates the property conjectures from user-defined benzenoids and properties in SageMath. Loading the CONJECTURING program also loads the function `propertyBasedConjecture()` which uses a list of benzenoids and properties to return automated conjectures. The program works to check combinations of properties on the list of known benzenoids in hopes of finding pairings of properties that hold for all known benzenoids. Adding more properties and more benzenoids in the program increases the complexity and number of automated conjectures.

Properties of benzenoids are often defined from numeric attributes of invariants. For instance, two properties can be defined from the parity of any invariant. Much of the

properties defined in this study are directly related to previously defined invariants from the earlier investigation. Before defining properties we will revisit the opening section on benzenoid embeddings. Embeddings are an important topic that allow us to give a graph theoretic interpretation of benzenoids. All benzenoids are defined as graphs comprised of vertex sets and an edge sets. A benzenoid is formed by closing a curve in the hexagonal lattice. The closed curve method is used to make sure the resulting graph is a valid benzenoid. We saw that defining benzenoids this way is useful for computing invariants. An invariant is a numeric property on the graph of a benzenoid that does not change based on a drawing. Two invariants that can be found for any graph G are its order: the number of vertices in G , and its size: the number of edges in G . In Section 1.3 we defined invariant functions for order and size as well as more invariants that are computable in any benzenoid. Properties are defined in the same way; they are definable for all benzenoids. A property-defined function returns a `True` truth value when a benzenoid has the defined property and returns a `False` truth value otherwise. Similar to an invariant-defined function, a property-defined function does not change truth value based on an embedding of a benzenoid.

In this property study we are still generating bounds for benzenoids. In general, bounds are often numeric. We generated and proved numeric upper bounds on the domination number of a benzenoid in Section 1.5. Those conjectures were formed by `CONJECTURING` from combinations of invariants supplied by us. Given a list of invariants, `CONJECTURING` systematically applies each invariant to a list of benzenoids to return a real number upper bound. Each conjecture claimed that the real number found through combinations of invariants would be larger than the domination number of the benzenoids known to `CONJECTURING`. If a benzenoid is found with domination number larger the conjectured upper bound, it is programmed in SageMath and added to the list of known benzenoids. As the list of known benzenoids grew, so did the accuracy of automated conjectures.

We take the idea of numeric bounds and extend it to bounds on properties of benzenoids. Numeric upper and lower bounds determined by invariants are equivalently upper and lower bounds determined by properties, given as *necessary* and *sufficient* conditions.

A necessary condition is a property that all benzenoids must have to be in a certain class. For instance, the assumption that a benzenoid B has an even number of vertices is a necessary condition for B to be in the class of benzenoids that all have equal cardinalities between the large and small bipartite sets. If a benzenoid does not have an even number of vertices, then there is no way its two disjoint bipartite sets can have the same number of elements. Supposing the necessary condition of even order is met is not enough to conclude the existence of bipartite sets with equal cardinalities. However, without having an even order, it is not possible to narrow the search in this subclass to find benzenoids with equal bipartite sets.

A sufficient condition is a property Q that ensures a benzenoid B will have some property P provided that B has property Q . The difference from a necessary condition is assuming B has condition Q and concluding this is enough for B to have property P as well. The next example uses the idea of bays and bay regions from Section 1.3. Let P be the property that benzenoid B contains a bay region and Q be the property that B contains a bay. Assuming B has property Q is enough to determine that B has property P . Having a bay is enough to conclude the existence of a bay region, since a bay is one type of bay region.

These bounds are structural since they provide a limit on classes a benzenoid may belong to. Necessary conditions are upper bounds. Using the example above, let P be the property that the larger and smaller bipartite sets of B have the same number of elements and let Q be the property that B has an even number of vertices. If a benzenoid has property P then it *necessarily* must have property Q as well. The relation between the properties is given by $P \subseteq Q$. Sufficient conditions are lower bounds. Using the previous

example, let P' be the property that B contains a bay region and Q' be the property that B contains a bay. Knowing that B has property Q' is *sufficient* to conclude it has property P' . The relation between the properties is given by $Q' \subseteq P'$.

We will generate property based conjectures using necessary and sufficient conditions. First, properties need to be defined in SageMath before CONJECTURING can begin returning automated conjectures.

2.1 Initial properties

Most of the following initial properties are definable all graphs. We define them for benzenoids. An invariant-defined function returns a numeric value and a property-defined function returns a Boolean truth value of True or False. It seems natural to begin the investigation in property conjectures by examining a benzenoid's vertices and edges. A graph is defined by a vertex set and an edge set. These two sets are usually the first pieces of information known about a graph. The number of vertices and the number of edges in a graph are invariants. These are integers which do not necessarily have constraints on how large or small they may be. We examined invariant conjectures on benzenoids, in particular on the domination number of a benzenoid, in Chapter 1.

A mathematical property differs from an invariant in that a property is either True or False. Often the defined properties are based on the numeric properties of invariants. Note that we are not generating conjectures on the number given by an invariant, but rather a property that number has. For instance, a property conjecture may be that an invariant is odd, or an invariant is not equal to zero. We first define properties on a benzenoid's order and size by their parity. For clarity, the following properties are written as program name first followed by a formal definition. The program code for each property may be found in Appendix A.

Property 1 (Even number of vertices). *Let B be a benzenoid of order $n(B)$. Define the property*

even_number_of_vertices(B) to return *True* when $n(B) = 2k$ for some $k \in \mathbb{Z}$ and return *False* otherwise.

Property 2 (Odd number of vertices). Let B be a benzenoid of order $n(B)$. Define the property *odd_number_of_vertices(B)* to return *True* when $n(B) = 2l + 1$ for some $l \in \mathbb{Z}$ and return *False* otherwise.

These properties can be programmed in SageMath with the help of previously defined Invariant 1, the order of a graph. Using the $B.order()$ method built into SageMath, we set the function *even_number_of_vertices(B)* to return *True* when $B.order()\%2==0$. The $\%$ symbol is predefined to be the modulus operator and will determine the remainder of dividing the order of B by 2. When the remainder is equal to 0, it indicates that $n(B)$ is even. In the same fashion, the function *odd_number_of_vertices(B)* returns *True* when $B.order()\%2==1$. In this case, division by 2 leaves a remainder of 1, so the order of B must be odd. Defining properties on the parity of a benzenoid's size is analogous.

Property 3 (Even number of edges). Let B be a benzenoid of size $m(B)$. Define the property *even_number_of_edges(B)* to return *True* when $m(B) = 2q$ for some $q \in \mathbb{Z}$ and return *False* otherwise.

Property 4 (Odd number of edges). Let B be a benzenoid of size $m(B)$. Define the property *odd_number_of_edges(B)* to return *True* when $m(B) = 2p + 1$ for some $p \in \mathbb{Z}$ and return *False* otherwise.

For determining the number of edges we use the provided $size()$ method in SageMath. Programming Properties 3 and 4 follows the same format as defining properties on order. When $B.size()\%2==0$ the function *even_number_of_edges(B)* returns *True*, and when $B.size()\%2==1$ the function *odd_number_of_edges(B)* returns *True*.

More properties are better than fewer when generating conjectures as more properties gives CONJECTURING more tools to work with. We may create another property relates

the number of vertices and the number of edges. The next property determines when a benzenoid has more edges than vertices.

Property 5 (More edges than vertices). *Let B be a benzenoid of order $n(B)$ and size $m(B)$. Define the property `more_edges_than_vertices(B)` to return `True` when $m(B) > n(B)$ and return `False` otherwise.*

It is helpful to use the `B.order()` and `B.size()` methods to define Property 5. Have SageMath compare `B.size()` and `B.order()` and return `True` when the B 's size is larger.

So far each property we defined has a truth value applicable to any arbitrary graph. Often it is a good idea to start a graph theoretic investigation with these universal properties and then narrow the scope to a specific class of graphs. Checking the properties defined so far using simple examples serves as a tool to check if the defined properties are working as intended. Clearly, it is important to make sure any programmed property code returns `True` or `False` when either result is expected.

Our primary interest in this study is to generate conjectures using properties applicable to benzenoids. The following properties are specifically attainable by benzenoids as opposed to any general graph. We return to the basic idea of embedding a benzenoid. In Section 1.3 we counted the number of hexagons, denoted $h(B)$ in a benzenoid embedding and returned that number as Invariant 13. Similar to order and size, we will define two properties based on the parity of $h(B)$.

Property 6 (Odd number of hexagons). *Let B be a benzenoid with $h(B)$ hexagons. Define the property `odd_number_of_hexagons(B)` to return `True` when $h(B) = 2s + 1$ for some $s \in \mathbb{Z}$ and return `False` otherwise.*

Property 7 (Even number of hexagons). *Let B be a benzenoid with $h(B)$ hexagons. Define the property `even_number_of_hexagons(B)` to return `True` when $h(B) = 2w$ for some $w \in \mathbb{Z}$ and return `False` otherwise.*

Programming the two properties in SageMath is straightforward to do if Invariant 13 is already defined. If not, use the equation proved in Theorem 1. The number of hexagons in B is given by $h(B) = m(B) - n(B) + 1$. For the property-defined function `even_number_of_hexagons(B)` we have SageMath return True when `hexagons(B)%2==0`. In this case, Invariant 13 leaves a remainder of 0 when divided by 2, so $h(B)$ must be even. The property-defined function `odd_number_of_hexagons(B)` works in the same way. Have SageMath return True when `hexagons(B)%2==1`. Then Invariant 13 leaves a remainder of 1 when divided by 2, so $h(B)$ is odd.

These properties are enough to use CONJECTURING and develop useful conjectures. The next section walks through using CONJECTURING in conjunction with SageMath to generate property conjectures for benzenoids. Using the properties we just defined, along with a few initial benzenoids, will be enough to begin proving conjectures and providing counterexamples.

2.2 The conjecture-making process

For the purposes of this paper, CONJECTURING may be thought of as a black box. We provide CONJECTURING with three inputs:

- Objects (in our case, graphs of benzenoids)
- Defined properties that assigns a truth value to each object
- A property of interest for which we want to conjecture on.

CONJECTURING takes these three inputs and returns conjectures which specifically involve the property of interest compared to other defined properties. A full description of how the program runs may be found in C. E. Larson & N. Van Cleemput [8] and an explanation of the working components is discussed in A. Bradford [1]. It is up to us to determine if a returned conjecture is true, or if there is a counterexample that refutes

the claim. The conjecture-making process begins by defining a few objects and properties. More objects and properties are added along the way. The starting point is up to us. We begin our investigation of benzenoid properties by first defining the properties stated in the previous section. In addition, we now define a property that determines if a benzenoid is *catacondensed*. Catacondensed benzenoids are defined and discussed in Section 1.4 of the invariant investigation. Reminder: a benzenoid is catacondensed when its inner dual is a tree.

There an equivalent definition of the catacondensed property that is can be defined explicitly without the need of defining an inner dual function. For any catacondensed benzenoid the number of internal vertices is 0. We can use this fact to define the catacondensed property. In this case, the property will check Invariant 14 (number of internal vertices) and Invariant 15 (number of external vertices) and examine their ratio. When the ratio between the two invariants is 0, it must be that there are 0 internal vertices.

Property 8 (Catacondensed). *Let B be a benzenoid with $n_i(B)$ internal vertices and $n_e(B)$ external vertices. Define the property $catacondensed(B)$ to return *True* when $\frac{n_i(B)}{n_e(B)} = 0$ and return *False* otherwise.*

To program the property in SageMath we take advantage of the already defined invariant functions `internal_vertices(B)` and `external_vertices(B)`. Checking the ratio between the two functions is enough to define Property 8. When the ratio equals 0, have Property 8 return *True*.

Table 2.1 contains a summary of the properties defined in the previous section. Add the catacondensed property into the mix. We continue the conjecture-making process by defining a couple of initial benzenoids.

The choice of initial benzenoids are arbitrary and up to personal preference. First let's use benzene, the only benzenoid comprised of one hexagon, and ground zero of the invariant investigation. In addition we will use triphenylene, a benzenoid which has overlap in all minimum dominating sets. Discussions on both overlap and minimum

Property	Summary
<code>even_number_of_vertices(B)</code>	$n(B) = 2k$ for some $k \in \mathbb{Z}$
<code>odd_number_of_vertices(B)</code>	$n(B) = 2l + 1$ for some $l \in \mathbb{Z}$
<code>even_number_of_edges(B)</code>	$m(B) = 2q$ for some $q \in \mathbb{Z}$
<code>odd_number_of_edges(B)</code>	$m(B) = 2p + 1$ for some $p \in \mathbb{Z}$
<code>more_edges_than_vertices(B)</code>	$m(B) > n(B)$
<code>odd_hexagons(B)</code>	$h(B) = 2s + 1$ for some $s \in \mathbb{Z}$
<code>even_hexagons(B)</code>	$h(B) = 2w$ for some $w \in \mathbb{Z}$

Table 2.1: Initial properties defined in Section 2.1

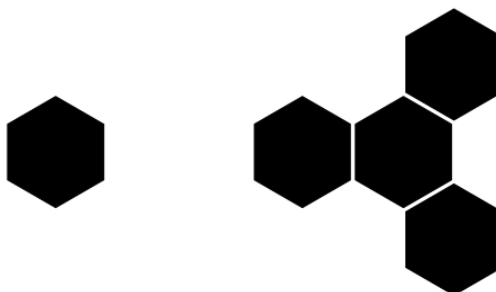


Figure 2.1: Benzene (left) and triphenylene (right)

dominating sets are in Section 1.2. The embeddings of benzene and triphenylene are shown in Figure 2.1.

As explained in the beginning of the section, CONJECTURING requires defined objects, defined properties, and a specific property to focus on. To start generating conjectures we begin by loading CONJECTURING into a SageMath worksheet along with a file containing defined objects and properties. The file containing objects and properties file should use the `.sage` extension, which is a SageMath notepad that does not contain executable cells. Since our objects are graphs, let's name the file `graphs_and_properties.sage`. Inside this file we define the two initial graphs along with Properties 1–7 from Section 2.1 and Property 8.

First the benzenoids are coded. The graphs of benzenoids are defined by an vertex set and an edge set. Once we start adding more benzenoids into the file, it quickly becomes cumbersome to define benzenoids with large order and size. For instance, to define benzene by its vertex set and edge set we would input the following code.

```
benzene = Graph(6)
benzene.add_edges([(0,2), (0,3), (2,4), (3,5), (5,1), (4,1)])
```

This tells SageMath to define benzene as a graph with six vertices where each vertex is assigned a number from the set $\{0, 1, 2, 3, 4, 5\}$. Then we use the SageMath method `add_edges()` on benzene to add the required edge needed to form the C_6 cycle that defines benzene. Triphenylene is defined with a vertex set and edge set in a similar fashion.

```
triphenylene = Graph(18)
triphenylene.add_edges([(0,2), (12,13), (15,14), (9,16), (3,8), (0,4), (2,5),
    (10,12), (16,17), (7,15), (5,11), (17,8), (4,9), (10,6), (9,1), (6,0), (5,1),
    (13,4), (7,2), (14,11), (1,3)])
```

It becomes clear to see that a benzenoid of significant size can take up space when defining it by its vertex set and edge set in this manner. In practice, we store a graph by its Graph6 string, which is an ASCII representation of a graph's adjacency matrix. The Graph6 encoding was developed by Brendan McKay from the Australian National University, and a full description may be found on his webpage¹. SageMath has the method `graph6_string()` already built in. Using benzene and triphenylene defined above, we run the methods on each benzenoid.

```
benzene.graph6_string()
triphenylene.graph6_string()
```

SageMath determines that the graphs benzene and triphenylene are encoded by the Graph6 strings 'ESX0' and 'QQ'c?0Q?C?_?AA?_?_A?C??_G', respectively. Then we copy the Graph6 strings for both benzenoids and store each in `graphs_and_properties.sage`.

```
benzene = Graph('ESX0')
```

¹<http://users.cecs.anu.edu.au/~bdm/data/formats.html>

```
triphenylene = Graph('QQ'c@?OQ?C?_?AA?_?_A@?C??_G')
```

Once both objects are defined we can define the properties from Table 2.1 and Property 8 into `graphs_and_properties.sage`. Constructing code that defines these properties requires a working knowledge of Python, a separate topic outside the scope of this investigation. Starting with the first property of Table 2.1, we encode `even_number_of_vertices` in our `.sage` file in the following way.

```
def even_number_of_vertices(B):  
    return B.order()%2==0
```

Here, the command `def` signals to SageMath that we want to define a new function named `even_number_of_vertices(B)` which takes a benzenoid `B` as an input. The colon indicates that the function is ready to be defined. The function we are defining is a property, and we return a Boolean truth value: either `True` or `False`. For a benzenoid to have an even number of vertices, it must be that its order is divisible by 2. This function determines the order of `B` with the method `B.order()` and uses modulo operator `%` to divide `B`'s order by 2. If the remainder of division by 2 is equal to 0, `even_number_of_vertices(B)` returns `True`. If the remainder of division by 2 is not equal to 0 then `even_number_of_vertices(B)` returns `False`. The other initial properties are defined as follows.

```
def odd_number_of_vertices(B):  
    return B.order()%2==1  
def even_number_of_edges(B):  
    return B.size()%2==0  
def odd_number_of_edges(B):  
    return B.size()%2==1  
def more_edges_than_vertices(B):  
    return (B.size() > B.order())
```

```
def odd_hexagons(B):  
    return hexagons(B)%2==1  
def even_hexagons(B):  
    return hexagons(B)%2==0
```

Notice that both `odd_hexagons(B)` and `even_hexagons(B)` call the invariant-defined function `hexagons(B)`, which uses the relation given in Theorem 1 to calculate the number of hexagons in benzenoid `B`. The `hexagons(B)` function is defined in Section 1.3 as Invariant 13 and should be included as well in `graphs_and_properties.sage` or property-defined functions `odd_hexagons(B)` and `even_hexagons(B)` will not execute.

```
def hexagons(B):  
    return B.size() - B.order() + 1
```

Finally we will program Property 8. We discussed earlier that a benzenoid is catacondensed when it has no internal vertices. We can import the functions determining the number of internal and external vertices from the previous investigation. The equations determining these two invariants are results of Theorem 2 and Corollary 1. The code for each of the invariants, as well as the catacondensed property is given below. Add all three to `graphs_and_properties.sage`.

```
def internal_vertices(B):  
    return 4*hexagons(B) + 2 - B.order()  
def external_vertices(B):  
    return B.order() - internal_vertices(B)  
def catacondensed(B):  
    return internal_vertices(B)/external_vertices(B)==0
```

Now the `graphs_and_properties.sage` file contains enough benzenoids and properties needed to begin generating automated conjectures. We proceed to find our first trial run of conjectures. Each trial is preformed in an individual cell of a SageMath work-

sheet. These worksheets use the `.sagews` file extension and are different from `.sage` files in that they contain executable cells. The code for the first conjecture run is given below, followed by an explanation.

```
load('graphs_and_properties.sage')
load('conjecturing.py')

benzenoids = [benzene, triphenylene]

properties = [even_number_of_vertices, odd_number_of_vertices,
              even_number_of_edges, odd_number_of_edges, more_edges_than_vertices,
              odd_hexagons, even_hexagons, catacondensed]

interest = properties.index(catacondensed)

conjectures = propertyBasedConjecture(benzenoids, properties, interest)

for c in conjectures:
    print c
```

In the cell above we first load `CONJECTURING` and well as `graphs_and_properties.sage` which contains the benzenoids and properties we just defined. Next we create a list, named `benzenoids` that acts as `CONJECTURING`'s list of known objects. Another list, named `properties` is created and informs `CONJECTURING` of the property-defined functions available from loading `graphs_and_properties.sage`. Next, `interest` is set to designate our specified "property of interest", which tells `CONJECTURING` to focus on generating conjectures that are bounds on this particular property. For these first conjectures we will set the `catacondensed` property as our property of interest. Then, the function `propertyBasedConjecture()`, which is defined in `conjecturing.py`, is called to be run

upon execution of the cell. The `propertyBasedConjecture()` function takes the list of benzenoids, list of properties, and a property of interest as inputs. These conjectures are returned as a list named `conjectures`. Finally, a for loop is called which iterates through the `conjectures` list and prints each conjecture on its own line. Every piece needed to generate conjectures is in place, and we may execute the SageMath cell.

Only one conjecture is returned in the first run. This is due to `CONJECTURING` working with only two benzenoids. As `CONJECTURING`'s list of known benzenoids increases, so will its list of returned conjectures.

It should be stated that for every automated conjecture there is a standard assumption that should always be assumed.

Standard Assumption. *Each automated conjecture is assumed to apply to all benzenoids.*

This first generated conjecture is returned in the form below.

Conjecture 7.

$$(even_number_of_vertices) \rightarrow (catacondensed)$$

We interpret this output as an "If P, then Q" statement. Explicitly stated: "For all benzenoids B, if B has property P then B has property Q". In the case of Conjecture 7 we have that `(even_number_of_vertices)` is our P property and the hypothesis. By extension, `(catacondensed)` is the Q property and the conclusion. Formally, this conjecture is stated as the following proposition.

Proposition. *Suppose a benzenoid B has $n(B)$ vertices. If $n(B)$ is even, then B is catacondensed.*

Now it is up to us to prove this conjecture, or find a counterexample that refutes it. Intuition tells us it is unlikely that every benzenoid with an even number of vertices has 0 internal vertices, a requirement to be catacondensed. The procedure to refute a conjecture is to find a benzenoid where the hypothesis is true, but the conclusion is false. It does not take long to find a counterexample to Conjecture 7.

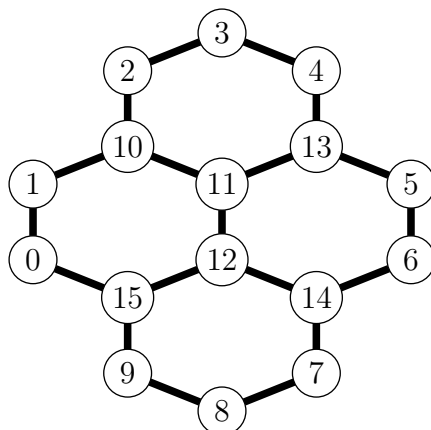


Figure 2.2: Pyrene, a counterexample to Conjecture 7

Disproof. Consider pyrene, shown in Figure 2.2. In this benzenoid, $n(B) = 16$. However, pyrene is not catacondensed as it contains 2 inner vertices. Figure 2.2 has a vertex labeling where vertices 11 and 12 are internal vertices. \square

We have determined Conjecture 7 is not true. What about the going the opposite direction? After a bit of thought, it turns out that the converse

$$(\text{catacondensed}) \rightarrow (\text{even_number_of_vertices})$$

is true. We have found our first property result for benzenoids! This gives way to our first property proposition.

Proposition 11. *Suppose a benzenoid B has $n(B)$ vertices. If B is catacondensed, then $n(B)$ is even.*

Proof. Let B be a benzenoid with $n(B)$ vertices and $h(B)$ hexagons. As B is catacondensed, it contains 0 internal vertices. Then $n_i(B) = 0$. By solving Theorem 2 for $n(B)$, we find that $n(B) = 4h(B) + 2$. It is clear to see that $n(B) = 4h(B) + 2 = 2(2h(B) + 1) = 2k$ for some $k \in \mathbb{Z}$ and it follows that $n(B)$ is even. \square

Once we find a counterexample that disproves a conjecture (in this case, pyrene is a counterexample to Conjecture 7) the next step is to add this graph to the benzenoids

list of objects supplied to CONJECTURING. This way we allow CONJECTURING to advance past the current conjecture with new information to generate new conjectures. To add pyrene we define a new graph using a vertex and edge set that represents pyrene in the plane. Then apply the `graph6_string()` method on the graph. Pyrene's Graph6 string is `'00'c'@G?_a???CA@_C??H'`. Define the graph pyrene using its Graph6 string in the file `graphs_and_properties.sage`.

After adding pyrene to the file of graphs and properties, copy the contents of the initial trial run in first SageMath cell into a new cell. Now update the list benzene to include pyrene. CONJECTURING's list of objects now reads `benzene = [benzene, triphenylene, pyrene]`. No further changes are needed before the cell may be executed for the next conjecture run. Activating the cell returns a new conjecture which CONJECTURING has determined to be a better bound on the catacondensed property given the addition of pyrene.

Conjecture 8.

$$(even_number_of_edges) \rightarrow (catacondensed)$$

Equivalently, Conjecture 8 is formally stated as the next proposition.

Proposition. *Suppose that B is a benzenoid with $m(B)$ edges. If $m(B)$ is even, then B is catacondensed.*

Again, intuition tells us that it is unlikely this true. That is okay, our goal is to refine CONJECTURING with many examples and properties. Finding counterexamples to generated conjectures gives the program more to work with. Our first hunch may be to examine different benzenoids with a small number of hexagons that contain at least 1 internal vertex. If a benzenoid has at least 1 internal vertex, it can not be catacondensed. Then we count the edges of each benzenoid and in hopes the benzenoid has an even amount. That would be enough to disprove the conjecture. The smallest benzenoid with containing an internal vertex is phenalene, which has a triangular structure and is shown



Figure 2.3: Phenalene



Figure 2.4: Fluoranthene, a counterexample to Conjecture 8

in Figure 2.3. Its inner dual is not a tree, and therefore not catacondensed. Upon further inspection, phenalene does not have an even number of edges as its size is 15. However, we can expand off of this benzenoid to create a new benzenoid with an even number of edges. Attaching a hexagon to phenalene such that the new hexagon is connected by a single edge will add 5 new edges to the graph, resulting in a benzenoid with a size of 20. This particular benzenoid is named fluoranthene, and its embedding is shown in Figure 2.4.

Disproof. Consider fluoranthene, shown in Figure 2.4, which has 20 edges and is not catacondensed. □

Note that the converse of Conjecture 8 is false as well. Each time a hexagon is added to a catacondensed benzenoid to create a new benzenoid, 5 edges are added. Each additional hexagon will flip the parity of $m(B)$ between even and odd. While Conjecture 8 and its converse did not reveal new benzenoid theory, we found a counterexample to add to CONJECTURING'S knowledge of benzenoids. Similar to the previous run, we find the

Graph6 string for fluoranthene, program the benzenoid in `graphs_and_properties.sage`, and then start another conjecture run.

The process of automated conjecture-making continues in this fashion. With each conjecture we either find a counterexample or a proof. Adding more properties into CONJECTURING gives the program a larger set of possible combinations to work with. In the next section we define more properties for benzenoids to generate more conjectures.

2.3 More properties

At this point we should define more properties into SageMath for CONJECTURING's use. Properties are the language that CONJECTURING uses to output conjectures. Each property expands the programs word bank and allows it to form new statements. More properties means more conjectures. We could define many properties that are applicable to all graphs. For instance, consider defining a property function that determines if a graph is connected or not. By construction all benzenoids are connected graphs. This property would be irrelevant to the investigation as the only objects we consider are benzenoids. The inclusion of this property in generated conjectures would be trivial.

The idea is to define more properties that separate benzenoids into classes. The properties should vary in truth value dependent on the benzenoid; it is not helpful to define properties that return True or False for *all* benzenoids. That is a reason for defining properties on the parity of an invariant. First we will define two properties on the domination number of a benzenoid. Determining upper bounds on the domination number was the goal of the invariant investigation. We will see results for property conjectures including the domination number as well. The next two properties check to see if the domination number of a benzenoid is even or odd.

Property 9 (Even domination number). *Let B be a benzenoid with domination number $\gamma(B)$. Define `even_domination_number(B)` to return `True` when $\gamma(B) = 2k$ for some $k \in \mathbb{Z}$ and*

return False otherwise.

Property 10 (Odd domination number). *Let B be a benzenoid with domination number $\gamma(B)$. Define `odd_domination_number(B)` to return `True` when $\gamma(B) = 2l + 1$ for some $l \in \mathbb{Z}$ and return `False` otherwise.*

These two properties can be defined with Invariant 3 (domination number), which uses the Python function `len()` on the SageMath built-in method `dominating_set()` to find the length of a minimum dominating set of a benzenoid. Since these two are property-defined functions, we have `even_domination_number(B)` return `True` when the domination number of B divided by 2 leaves a remainder of 0. In the same way, we set `odd_domination_number(B)` return `True` when the domination number of B divided by 2 leaves a remainder of 1.

The next property compares the domination number of a benzenoid to its hexagon count. When a benzenoid has the same domination number as the number of hexagons in its embedding, we say that the benzenoid has *equality*. Equivalently, Invariant 13 returns the same integer as the one returned by Invariant 3. That is, $h(B) = \gamma(B)$.

Property 11 (Equality). *Let B be a benzenoid with $h(B)$ hexagons and domination number $\gamma(B)$. Define `equality(B)` to return `True` when $h(B) = \gamma(B)$ and return `False` otherwise.*

Code this property by calling the invariant functions for number of hexagons and domination number. The property-defined function `equality(B)` will return `True` after checking that `hexagons(B)==domination_number(B)` for any B that has this equality between the two invariants.

The following properties are similar to the one above in their methodology. Each property is based on when two invariants are equal. First we find benzenoids that have an equal number of internal and external vertices. The property compares Invariant 14 and Invariant 15.

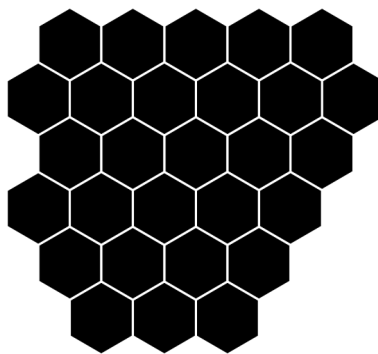


Figure 2.5: Benzenoid B with Property 12: $n_i(B) = n_e(B) = 36$

Property 12 (Equal internal and external vertices). *Let B be a benzenoid with $n_i(B)$ internal vertices and $n_e(B)$ external vertices. Define `equal_internal_and_external_vertices(B)` to return `True` when $n_i(B) = n_e(B)$ and return `False` otherwise.*

Note that a benzenoid with the above property will have a larger order than the examples given so far. Figure 2.5 shows one example of a benzenoid with equal internal and external vertices. Defining the property is straightforward provided Invariants 14 and 15 are already defined. Then coding the property just involves determining if the ratio between internal vertices and external vertices is 1. That is, a benzenoid B will have Property 12 when $\frac{n_i(B)}{n_e(B)} = 1$. In SageMath, this ratio is programmed as `internal_vertices(B)/external_vertices(B)==1`.

Whenever the ratio $\frac{n_i(B)}{n_e(B)} \neq 0$ means benzenoid B has at least 1 internal vertex. The following property determines if B contains at least 1 internal vertex by checking an aforementioned invariant-defined function.

Property 13 (Contains internal vertices). *Let B be a benzenoid with $n_i(B)$ internal vertices. Define `contains_internal_vertices(B)` to return `True` when $n_i(B) \neq 0$ and return `False` otherwise.*

This property is easy to program in SageMath. We make use of previously defined Invariant 14 from Section 1.3. Define Property 13 to check that that the invariant-defined function is not equal to 0 with the line `internal_vertices(B)!=0`. When it is indeed the

case that $n_i(B) \neq 0$, Property 13 returns True.

Now we will compare the cardinality of a benzenoid's bipartite sets. The cardinality of the smaller bipartite set, denoted $|\mathcal{B}_s(B)|$, is defined as Invariant 10. Invariant 11 defines the cardinality of the larger bipartite set, denoted $|\mathcal{B}_l(B)|$. In Section 1.3 we mentioned that a benzenoid is *balanced* when $|\mathcal{B}_s(B)| = |\mathcal{B}_l(B)|$. The property is now defined below.

Property 14 (Balanced). *Let B be a benzenoid with smaller bipartite set $\mathcal{B}_s(B)$ and larger bipartite set $\mathcal{B}_l(B)$. Define `balanced(B)` to return `True` when $|\mathcal{B}_s(B)| = |\mathcal{B}_l(B)|$ and return `False` otherwise.*

In addition, a benzenoid that has the strict inequality $|\mathcal{B}_l(B)| > |\mathcal{B}_s(B)|$ is said to have color excess. A benzenoid B belongs to one of two disjoint sets: either B is in the set of benzenoids that are balanced, or B is in the set of benzenoids with color excess.

Property 15 (Contains color excess). *Let B be a benzenoid with smaller bipartite set $\mathcal{B}_s(B)$ and larger bipartite set $\mathcal{B}_l(B)$. Define `contains_color_excess(B)` to return `True` when $|\mathcal{B}_l(B)| > |\mathcal{B}_s(B)|$ and return `False` otherwise.*

A straightforward way to program Property 15 is to use previously defined Invariant 12. In SageMath, check that `color_excess(B) != 0`. If the invariant-defined function is not equal to 0, return True.

At the end of Section 1.3 we defined an invariant that counts the number of bay regions in a benzenoid. There are four bay regions possible in a benzenoid: fissures, bays, coves, and fjords. Figure 1.7 shows a benzenoid that has one of each bay region. First we define a property that determines if a benzenoid has at least one bay region.

Property 16 (Contains bay region). *Let B be a benzenoid with $b(B)$ bay regions. Define `contains_bay_region(B)` to return `True` when $b(B) > 0$ and return `False` otherwise.*

The number of bay regions is defined as Invariant 16. Then we will program the function `contains_bay_region(B)` to call the invariant function `number_of_bay_regions(B)`

and check if the integer returned by the invariant is larger than 0. If this is the case, Property 16 will return `True`. We continue by defining properties for each type of bay region. Programming each of the bay region properties is straightforward if the bay region invariant-defined functions are carried over from Section 1.3.

Property 17 (Contains fissures). *Let B be a benzenoid. Define `contains_fissures(B)` to return `True` when B contains at least one fissure and return `False` otherwise.*

Note that benzene is the only benzenoid that does not contain a fissure. Have SageMath use Invariant 17 to determine if the number of fissures is not equal to 0. This check is programmed in Python as `fissures(B) != 0`.

Property 18 (Contains bays). *Let B be a benzenoid. Define `contains_bays(B)` to return `True` when B contains at least one bay and return `False` otherwise.*

Similar to Property 17, the above function checks if Invariant 18 is not equal to 0. In SageMath, this check is coded as `bays(B) != 0`. The next two bay region properties are defined and programmed in a similar manner.

Property 19 (Contains coves). *Let B be a benzenoid. Define `contains_coves(B)` to return `True` when B contains at least one bay and return `False` otherwise.*

Property 20 (Contains fjords). *Let B be a benzenoid. Define `contains_fjords(B)` to return `True` when B contains at least one fjord and return `False` otherwise.*

For Property 19 we define a function in SageMath that checks Invariant 19 to determine if `coves(B) != 0`. Similarly, for Property 20, a function is defined in SageMath that determines if `fjords(B) != 0` using Invariant 20. In either case, if the two invariants are found to not equal 0, both property functions will return `True`.

The final properties defined in this section are based on Invariants 24–29 (number of 0–5 external edges) having values greater than 0. These property-defined functions determine if a benzenoid contains a hexagon with varying amounts of external edges.

Property 21 (Contains hexagon with 0 external edges). Let B be a benzenoid. Define *contains_face_with_0_external_edges*(B) to return *True* when B contains at least one hexagon with 0 external edges and return *False* otherwise.

Property 22 (Contains hexagon with 1 external edge). Let B be a benzenoid. Define *contains_face_with_1_external_edge*(B) to return *True* when B contains at least one hexagon with 1 external edge and return *False* otherwise.

Property 23 (Contains hexagon with 2 external edges). Let B be a benzenoid. Define *contains_face_with_2_external_edges*(B) to return *True* when B contains at least one hexagon with 2 external edges and return *False* otherwise.

Property 24 (Contains hexagon with 3 external edges). Let B be a benzenoid. Define *contains_face_with_3_external_edges*(B) to return *True* when B contains at least one hexagon with 3 external edges and return *False* otherwise.

Property 25 (Contains hexagon with 4 external edges). Let B be a benzenoid. Define *contains_face_with_4_external_edges*(B) to return *True* when B contains at least one hexagon with 4 external edges and return *False* otherwise.

Property 26 (Contains hexagon with 5 external edges). Let B be a benzenoid. Define *contains_face_with_5_external_edges*(B) to return *True* when B contains at least one hexagon with 5 external edges and return *False* otherwise.

Properties 21–26 are defined using Invariants 24–29. Each property-defined function calls an corresponding invariant-defined function *faces_with_i_external_edges*(B) for $i \in \{0, 1, 2, 3, 4, 5\}$ and checks if the invariant-defined function returns 0.

Armed with these properties, and a list of benzenoids, CONJECTURING has enough to work with to begin returning interesting conjectures. Next we discuss techniques for constructing counterexamples to conjectures. Following this, we will state and prove conjectures generated by CONJECTURING.

2.4 Clustering

This section explores methods of finding counterexamples to automated conjectures. The process of methodically finding counterexamples is called *clustering*, which refers to building a counterexample from an already defined benzenoid. Familiarization with properties common in many benzenoids often makes it easy to guess ahead of time if certain automated conjectures are provable or if counterexamples exist. That is, if we conjecture that a generated conjecture has a counterexample, we often have an idea of how to construct such a counterexample. This is a benefit that allows optimized counterexample construction through pattern recognition. These counterexamples provide more knowledge to CONJECTURING, which refines the accuracy of automated conjectures. First clustering is studied with *balanced* benzenoids, defined as Property 14 in the previous section.

A counterexample to a balanced conjecture (meaning balanced is specified as the property of interest in CONJECTURING) is a benzenoid with color excess. In Section 1.4 we examined triangulenes, a class of benzenoids with pyramidal structure. These pyramidal benzenoid are likely candidates to contain color excess (defined as Property 15) based on observations in [3]. In their paper, Cyvin & Gutman prove the following theorem that provides an alternate way to calculate color excess by subtracting the number of *valleys* in a benzenoid from its number of *peaks*. Peaks are vertices at the highest point in a benzenoid's embedding. Valleys are vertices at the lowest point of the embedding.

Theorem 4 (Cyvin & Gutman [3]). *Let B be a benzenoid with $n_{\wedge}(B)$ peaks and $n_{\vee}(B)$ valleys. Then the color excess of B is given by $\triangleleft(B) = |n_{\wedge}(B) - n_{\vee}(B)|$.*

Theorem 4 gives us the relation $\triangleleft(B) = |\# \text{ of peaks} - \# \text{ of valleys}|$ which will be useful for finding counterexamples. Figure 2.6 shows an example color excess calculation with green peaks and purple valleys.

Triangulenes by construction always have one single peak and many more valleys.

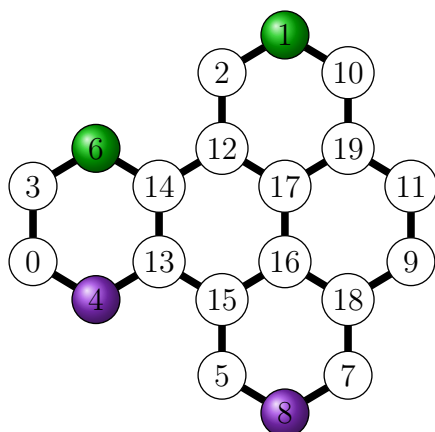


Figure 2.6: $\langle(B) = |\# \text{ of peaks} - \# \text{ of valleys}| = |2 - 2| = 0$

The only triangulene that does not contain color excess is benzene. Let's use triangulenes, and similar benzenoids, to create counterexamples to automated conjectures. Here is one automated conjecture with balanced specified as the property of interest.

Conjecture 9.

$$(even_number_of_vertices) \rightarrow (balanced)$$

Proposition. *Suppose B is a benzenoid with an even number of vertices. Then B is balanced.*

Finding a triangulene with an even number of vertices will be enough to disprove Conjecture 9. Not much searching is needed as triangulene-3 has the even order we are looking for.

Disproof. Consider triangulene-3, denoted T_3 . Figure 2.7 shows one labeling of T_3 . Benzenoid T_3 has 22 vertices, and so its order is even. However, T_3 is not balanced as $\langle(T_3) = |\# \text{ of peaks} - \# \text{ of valleys}| = |1 - 3| = 2$, and B contains color excess by Theorem 4. □

Figure 2.8 shows the peaks and valleys argument for T_3 . We proceed in the conjecture-making process by defining triangulene-3 in SageMath and then adding it to the list of objects in CONJECTURING. Next more conjectures are generated. One of the returned conjectures in the new batch is given below.

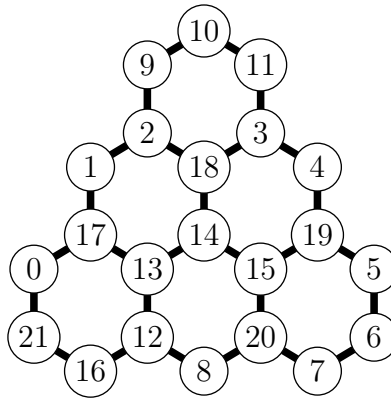


Figure 2.7: A vertex labeling of triangulene-3

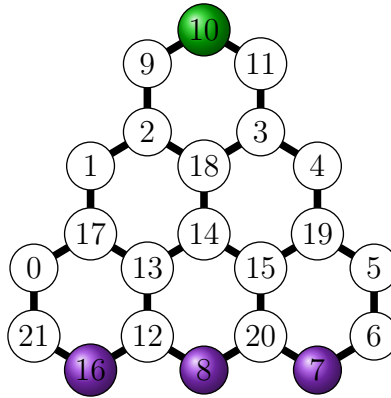


Figure 2.8: Color excess in triangulene-3: $\langle(T_3) = 2$

Conjecture 10.

$$((\text{odd_number_of_edges}) \& (\text{even_number_of_vertices})) \& (\text{even_domination_number}) \rightarrow (\text{balanced})$$

Proposition. *Let B be a benzenoid. Suppose B has an odd number of edges, an even number of vertices, and an even domination number. Then B is balanced.*

A counterexample to Conjecture 10 may be constructed by attaching a pyramidal cluster to a benzenoid that already has the properties specified by the hypothesis. Some manipulation may be involved to ensure the hypothesis is still met in the resulting benzenoid, but if the hypothesis can be met there likely exists a pyramidal cluster large enough to disprove the conjecture.

Disproof. Let B be the benzenoid shown in Figure 2.9. Note that $m(B) = 63$, $n(B) = 50$,

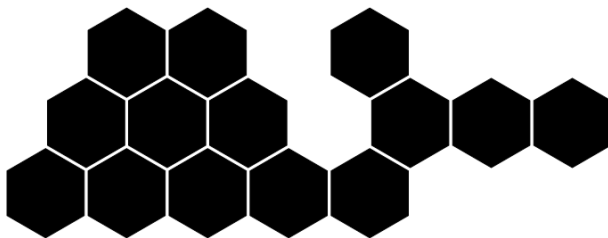


Figure 2.9: A counterexample to Conjeture 10

and $\gamma(B) = 14$, so B meets the assumptions given in the hypothesis. However, B is not balanced. Observe that $\triangleleft(B) = |\# \text{ of peaks} - \# \text{ of valleys}| = 2$. By Theorem 4, B contains color excess. □

The takeaway when generating conjectures on the property balanced is that, in many cases, attaching a pyramidal cluster to a benzenoid meeting the hypothesis is enough to find a counterexample. The next property we will apply clustering methods on are benzenoids containing *perfect matchings*, which is yet to be defined. Recall from Section 1.3 that a *matching* of a graph G is a set of independent edges from $E(G)$. A perfect matching is explained in [2] as a matching M of graph G with the property that every vertex from $V(G)$ is incident to exactly one edge in the matching M . We can use perfect matchings to create more benzenoid classes.

Property 27 (Contains perfect matching). *Let B be a benzenoid and M be a matching of B . Define `contains_perfect_matching(B)` to return `True` when M is a perfect matching of B and return `False` otherwise.*

It is fine to state this property, but the definition does not provide a programming method. To code this property we use the fact that every perfect matching of benzenoid B must be of size $\frac{n(B)}{2}$. Every edge has two endpoints, and a perfect matching must contain all vertices. As it is a matching, the edges must independently cover B . Adding one more edge to a perfect matching would not be a matching at all. Then a perfect matching is a maximal matching. We can define Property 27 using Invariant 9, the matching number. For benzenoid B , defined in SageMath, code Property 27 to return

the truth value of the equality check $\text{matching_number}(B) == B.\text{order}()/2$.

Now set `contains_perfect_matching` as the property of interest in CONJECTURING and start a conjecture run. The next conjecture comes from the first batch.

Conjecture 11.

$$(\text{balanced}) \rightarrow (\text{contains_perfect_matching})$$

Proposition. *Let B be a benzenoid. If B is balanced, then B contains a perfect matching.*

It is not immediately apparent if the proposition is true or false. One strategy to try proving the proposition would be to show that, for a balanced benzenoid B , any subset $S \subseteq V(B)$ has the property that the cardinality of the neighboring set of S is greater than or equal to the cardinality of S . If this could be shown, we could use Hall's Marriage Theorem to conclude that B has a perfect matching. This theorem, proven in 1935 by Philip Hall, is given below.

Theorem 5 (Hall's Marriage Theorem). *Suppose G is a bipartite graph with bipartition (A, B) . Then G has a perfect matching if and only if $|A| = |B|$ and for any $S \subseteq V(G)$, $|N_G(S)| \geq |S|$.*

Given that benzenoids have a nice structure, it seems reasonable to show that all balanced benzenoids have this neighborhood requirement. However, constructing a proof of this would be in vain, as Lovász & Plummer [11] give an example of a graph that happens to be a balanced benzenoid, yet does not have a perfect matching. This benzenoid, shown in Figure 2.10, is enough to refute Conjecture 11.

Disproof. Let B' be the benzenoid shown in Figure 2.10 and let $\mu(B')$ denote the matching number of B' . Observe that B' is balanced, since $\triangleleft(B') = |\# \text{ of peaks} - \# \text{ of valleys}| = 0$. However, B' does not contain a perfect matching as $\mu(B') = 20 \neq 21 = \frac{n(B')}{2}$. \square

Add the counterexample benzenoid to the list of objects known to CONJECTURING. The strategy will be to use this counterexample to expand from on future conjectures on

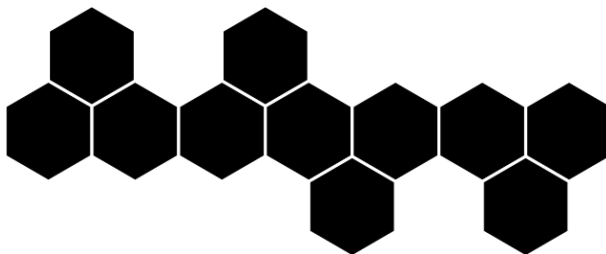


Figure 2.10: Benzenoid B': Figure 1.1.1 in Lovász & Plummer [11]

the `contains_perfect_matching` property. With the updated list of benzenoids, the next conjecture adds Property 21 in the hypothesis.

Conjecture 12.

$$((\text{contains_face_with_0_external_edges}) \& (\text{balanced})) \rightarrow (\text{contains_perfect_matching})$$

Proposition. *Let B be a benzenoid. If B contains a hexagon with 0 external edges and B is balanced, then B contains a perfect matching.*

Knowing B' does not contain a perfect matching is enough to construct a counterexample to Conjecture 12. We proceed by adding hexagons. Attaching a circulene to B' will be enough to disprove the conjecture. Circulenes are a class of benzenoids that contain at least 1 hexagon with 0 external edges, excluding the trivial circulene benzene. Examples of circulenes are shown in Figure 1.13 of Section 1.4.

Disproof. Let B be the benzenoid shown in Figure 2.11 with matching number $\mu(B)$. Then B contains a hexagon with 0 external edges and is balanced as $\langle(B) = |\# \text{ of peaks} - \# \text{ of valleys}| = 0$. However, B does not contain a perfect matching as $\mu(B) = 31 \neq 32 = \frac{n(B)}{2}$. □

Continue the conjecture-making process by defining this benzenoid in SageMath and adding the graph to the list of known objects in CONJECTURING. In the next conjecture, the hypothesis is updated again to include Property 9.

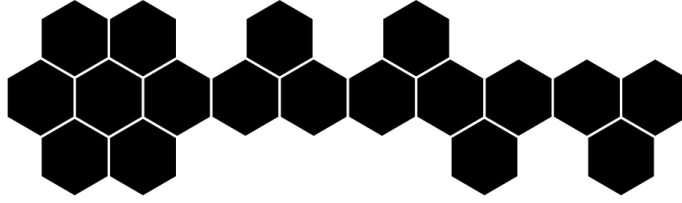


Figure 2.11: A counterexample to Conjecture 12

Conjecture 13.

$$(((balanced) \& (even_domination_number)) \& (contains_face_with_0_external_edges)) \rightarrow (contains_perfect_matching)$$

Proposition. *Let B be a benzenoid. If B is balanced, has an even domination number, and contains at least one hexagon with 0 external edges, then B contains a perfect matching.*

The difference between Conjectures 12 and 13 is the extra condition that B must have an even domination number. Attaching a hexagon to a e_{22} edge on the perimeter of the benzenoid shown in Figure 2.11 increases the domination number by 1, enough to disprove the conjecture.

Disproof. Let B be the benzenoid shown in Figure 2.12 with domination number $\gamma(B)$ and matching number $\mu(B)$. Note that B is balanced as $\triangleleft(B) = |\# \text{ of peaks} - \# \text{ of valleys}| = 0$. Further, $\gamma(B) = 20$, which is even, and B contains a hexagon with 0 external edges. However, B does not contain a perfect matching, as $\mu(B) = 33 \neq 34 = \frac{n(B)}{2}$. \square

Adding this benzenoid to objects known to CONJECTURING increases the program's knowledge and refines conjectures by increasing the complexity. In the next conjecture run another property is added to the mix. The process is still the same. As long as B' is part of the cluster we can continue to find counterexamples for many generated conjectures.

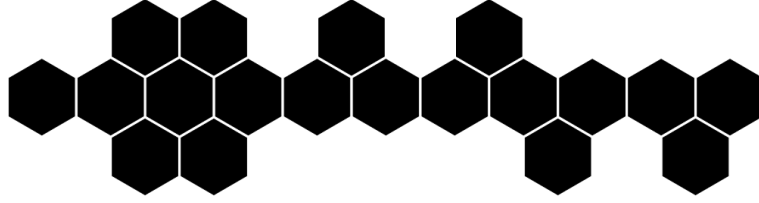


Figure 2.12: A counterexample to Conjecture 13

2.5 Results

Up to this point, CONJECTURING has generated sufficient condition conjectures that are lower bounds for the user-specified property of interest. By default, the sufficient parameter in function `propertyBasedConjecture()` is set to `True`. Changing the value to `False` returns necessary condition conjectures that are upper bounds for the property of interest. The results shown here include both necessary and sufficient condition conjectures for various properties of interest. We present the results in a conjecture-proposition-proof format. First, the generated conjecture is shown. Next, a proposition equivalent to the conjecture is formally stated, followed by a proof.

Conjecture 14.

$$(balanced) \rightarrow (even_number_of_vertices)$$

Proposition 12. *If a benzenoid B is balanced, then B has an even number of vertices.*

Proof. Let $\mathcal{B}_s(B)$ be the smaller bipartite set of B and let $\mathcal{B}_l(B)$ be the larger bipartite set of B . As B is balanced, $|\mathcal{B}_s(B)| = |\mathcal{B}_l(B)|$. Then the number of vertices in B is given by

$$\begin{aligned} n(B) &= |\mathcal{B}_s(B)| + |\mathcal{B}_l(B)| \\ &= |\mathcal{B}_s(B)| + |\mathcal{B}_s(B)| \\ &= 2|\mathcal{B}_s(B)|. \end{aligned}$$

Let $k = |\mathcal{B}_s(B)|$ for some $k \in \mathbb{Z}$. Then $n(B) = 2k$ and by definition B has an even number of vertices. □

Conjecture 15.

$$(catacondensed) \rightarrow (balanced)$$

Proposition 13. *Let B be a benzenoid with $h(B)$ hexagons. If B is catacondensed, then B is balanced.*

Proof. We prove this by induction on the number of hexagons in B . In the base case when $h(B) = 1$, B is benzene. Benzene is catacondensed, as it has 0 internal vertices, and balanced, as $|\mathcal{B}_s(B)| = |\mathcal{B}_l(B)|$. Suppose the assumption that whenever B is catacondensed then B is balanced holds for benzenoids with $k - 1 \geq 1$ hexagons, where $k \in \mathbb{N}$. We will show the assumption holds for k hexagons. Let B be a catacondensed benzenoid with k hexagons. By Lemma 1 there exists a removable hexagon H in B . Define $B' = B - H$ to be the benzenoid obtained by removing H from B . As B is catacondensed, removing H from B eliminates 4 vertices from $V(B)$. These observations show the two relations $h(B') = h(B) - 1 = k - 1$ and $n(B') = n(B) - 4$.

By the inductive hypothesis, B' is balanced. Then $|\mathcal{B}_s(B')| = |\mathcal{B}_l(B')|$. Let v and w denote the 2 vertices on the perimeter of B' that are incident to an edge in H . Without loss of generality, suppose $v \in \mathcal{B}_s(B)$ and $w \in \mathcal{B}_l(B)$. The proof is analogous when $v \in \mathcal{B}_l(B)$ and $w \in \mathcal{B}_s(B)$. Attach H to the vw edge to form B . Denote the 4 vertices that are in $V(B)$ and not in $V(B')$ as a, b, c , and d . Again, without loss of generality, let va, ab, bc, cd , and dw be edges in $E(B)$ that necessarily are not in $E(B')$. As $v \in \mathcal{B}_l(B)$, vertex a must belong to smaller bipartite set $\mathcal{B}_s(B)$. In an alternating method, $b \in \mathcal{B}_l(B), c \in \mathcal{B}_s(B)$, and $d \in \mathcal{B}_l(B)$.

We have that $|\mathcal{B}_s(B)| = |\mathcal{B}_s(B')| + 2$ and $|\mathcal{B}_l(B)| = |\mathcal{B}_l(B')| + 2$. Observe that

$$\begin{aligned} |\mathcal{B}_s(B)| &= |\mathcal{B}_s(B')| + 2 \\ &= |\mathcal{B}_l(B')| + 2 \\ &= |\mathcal{B}_l(B)|, \end{aligned}$$

and by definition B is balanced. It follows by mathematical induction that whenever benzenoid B is catacondensed, B is balanced as well. \square

Conjecture 16.

$$(catacondensed) \rightarrow (\sim (contains_face_with_0_external_edges))$$

Proposition 14. *Let B be a catacondensed benzenoid. Then B does not contain a hexagon with 0 external edges.*

Proof. A hexagon with 0 external edges must have 6 internal edges. In this case each edge connects to another hexagon. Hexagons surrounding a hexagon with 0 external edges form a cycle of length 6 in the benzenoid's corresponding inner dual. Benzenoid B may not have such a cycle in its inner dual as, by assumption, B is catacondensed. By definition of catacondensed, the inner dual of B is a tree. It follows that when B is catacondensed, B does not contain a hexagon with 0 external edges. \square

Conjecture 17.

$$(odd_number_of_vertices) \rightarrow (contains_color_excess)$$

Proposition 15. *Let B be a benzenoid with an odd number of vertices. Then B contains color excess.*

Proof. As B had an odd number of vertices, the smaller bipartite set and larger bipartite set must have different cardinalities. Then $|\mathcal{B}_l(B)| > |\mathcal{B}_s(B)|$. Subtract both sides of the inequality by $|\mathcal{B}_s(B)|$. Then

$$|\mathcal{B}_l(B)| - |\mathcal{B}_s(B)| > 0.$$

Since the difference between the cardinalities of the larger bipartite set and smaller bipartite set is greater than 0, benzenoid B contains color excess. \square

Conjecture 18.

$$(equality) \rightarrow (contains_internal_vertices)$$

Proposition 16. *Let B be a benzenoid with $h(B)$ hexagons and domination number $\gamma(B)$. If $h(B) = \gamma(B)$, then B contains internal vertices.*

Proof. We will prove this proposition by showing the contrapositive. Let B be a benzenoid that does not contain internal vertices. Then B is catacondensed. We proceed by using mathematical induction on $h(B)$. When $h(B) = 1$, benzenoid B is benzene, which has a domination number of $\gamma(B) = 2$. Then $h(B) \neq \gamma(B)$ and the base case holds.

Suppose the assumption that if B does not contain internal vertices, then $h(B) \neq \gamma(B)$ holds for $k - 1 \geq 1$ hexagons, where $k \in \mathbb{N}$. We will show the assumption holds for k hexagons. Let B be a benzenoid with 0 internal vertices and $h(B) = k$ hexagons. By Lemma 1, there exists a removable hexagon H in B . Let $B' = B - H$ be the benzenoid defined by removing H from B . Removing H removes 4 vertices. Each vertex in a minimum dominating set may cover at most 4 vertices. Then removing 4 vertices from B will reduce $\gamma(B)$ by at least 1 or at most 2. Suppose the minimal case where $\gamma(B)$ is reduced by 1. If the induction holds in the minimal case it will hold in the maximal case. Now we have the two relations $h(B') = h(B) - 1 = k - 1$ and $\gamma(B') = \gamma(B) - 1$.

As $h(B') = k - 1$, we may apply the inductive hypothesis. Then $h(B') \neq \gamma(B')$. Attach H to B' to form B . From the equations above, $h(B) = h(B') + 1$ and $\gamma(B) = \gamma(B') + 1$. Observe that

$$h(B) = h(B') + 1 \neq \gamma(B') + 1 = \gamma(B),$$

and we find that $h(B) \neq \gamma(B)$. It follows by mathematical induction and proof by contrapositive that if $h(B) = \gamma(B)$, then benzenoid B contains internal vertices. \square

Conjecture 19.

$$(catacondensed) \rightarrow (\sim (equality))$$

Proposition 17. *Let B be a benzenoid. If B is catacondensed, then $\gamma(B) \neq h(B)$.*

Proof. Consider the contrapositive of the statement. Let B be a benzenoid that contains equality. That is $\gamma(B) = h(B)$. By Proposition 16, as B contains equality, B also contains internal vertices. Then $n_i(B) > 0$ where $n_i(B)$ is the number of internal vertices in B. Catacondensed benzenoids contain 0 internal vertices. As $n_i(B) > 0$, benzenoid B is not catacondensed, proving the contrapositive. \square

Conjecture 20.

(equal_internal_and_external_vertices) \rightarrow (even_number_of_vertices)

Proposition 18. *Let B be a benzenoid. If B has an equal number of internal and external vertices, then B has an even number of vertices.*

Proof. Let $V_e(B)$ be the set of external vertices of B and $V_i(B)$ be the set of internal vertices of B. The union of sets $V_e(B)$ and $V_i(B)$ is the vertex set of B, so the order of B is given by $n(B) = |V_e(B)| + |V_i(B)|$. By assumption, $|V_e(B)| = |V_i(B)|$. Using substitution,

$$\begin{aligned} n(B) &= |V_e(B)| + |V_i(B)| \\ &= |V_e(B)| + |V_e(B)| \\ &= 2|V_e(B)|. \end{aligned}$$

Let $m = |V_e(B)|$ for some $m \in \mathbb{Z}$. Then $n(B) = 2m$ and by definition B contains an even number of vertices. \square

Conjecture 21.

((odd_number_of_edges) \wedge (odd_number_of_vertices)) \rightarrow (even_hexagons)

Proposition 19. *Suppose B is a benzenoid where:*

1. B has an odd number of edges and an even number of vertices or
2. B has an even number of edges and an odd number of vertices.

Then B has an even number of hexagons.

Proof. Case 1. Suppose $m(B) = 2k + 1$ and $n(B) = 2l$ for some $k, l \in \mathbb{Z}$. By Theorem 1,

$$h(B) = m(B) - n(B) + 1 = 2k + 1 - 2l + 1 = 2(k - l + 1).$$

As k and l are integers, $k - l + 1 \in \mathbb{Z}$ as well. Let $a = k - l + 1$. Then $h(B) = 2a$ and by definition $h(B)$ is even.

Case 2. Now suppose $m(B) = 2g$ and $n(B) = 2f + 1$ for some $g, f \in \mathbb{Z}$. By Theorem 1,

$$h(B) = m(B) - n(B) + 1 = 2g - (2f + 1) + 1 = 2(g - f).$$

As g and f are integers, $g - f \in \mathbb{Z}$ as well. Let $b = g - f$. Then $h(B) = 2b$ and by definition $h(B)$ is even. In both cases we have that benzenoid B contains an even number of hexagons, which is what we needed to show. \square

Conjecture 22.

$$((equality) \& (odd_hexagons)) \rightarrow (odd_domination_number)$$

Proposition 20. *Let B be a benzenoid with $h(B)$ hexagons and domination number $\gamma(B)$. Suppose $h(B) = \gamma(B)$ and B has an odd number of hexagons. Then B has an odd domination number.*

Proof. As B has an odd number of hexagons, $h(B) = 2k + 1$ for some $k \in \mathbb{Z}$. By assumption, $h(B) = \gamma(B)$. Then $\gamma(B) = 2k + 1$, and by definition $\gamma(B)$ is odd. It follows B has an odd domination number. \square

Conjecture 23.

$$((equality) \& (even_hexagons)) \rightarrow (even_domination_number)$$

Proposition 21. *Let B be a benzenoid with $h(B)$ hexagons and domination number $\gamma(B)$. Suppose $h(B) = \gamma(B)$ and B has an even number of hexagons. Then B has an even domination number.*

Proof. The proof is analogous to Proposition 20. By assumption, B has an even number of hexagons and by definition $h(B) = 2l$ for some $l \in \mathbb{Z}$. Also by assumption, $h(B) = \gamma(B)$. Then $2l = h(B) = \gamma(B)$, and by definition B has an even domination number. \square

Conjecture 24.

$$(contains_perfect_matching) \rightarrow (balanced)$$

Proposition 22. *Let B be a benzenoid. Suppose B contains a perfect matching, then B is balanced.*

Proof. Suppose benzenoid B contains a perfect matching. By Theorem 5 (Hall's Marriage Theorem), if B has a perfect matching then, in particular, $|\mathcal{B}_l(B)| = |\mathcal{B}_s(B)|$ for bipartite sets $\mathcal{B}_l(B)$ and $\mathcal{B}_s(B)$. By definition, B is balanced. \square

Conjecture 25.

$$(contains_color_excess) \rightarrow (\sim (contains_perfect_matching))$$

Proposition 23. *Let B be a benzenoid. Suppose B contains color excess, then B does not contain a perfect matching.*

Proof. Consider the contrapositive of this statement. Suppose B contains a perfect matching. We want to show B does not contain color excess. By Proposition 22, as B contains a perfect matching, B must be balanced. Then $|\mathcal{B}_s(B)| = |\mathcal{B}_l(B)|$. Notice that $|\mathcal{B}_l(B)| - |\mathcal{B}_s(B)| = 0$, and by definition B does not contain color excess. \square

Conjecture 26.

$$(catacondensed) \rightarrow (contains_perfect_matching)$$

Proposition 24. *Let B be a benzenoid. If B is catacondensed, then B contains a perfect matching.*

Proof. We will prove this statement using mathematical induction. When $h(B) = 1$, benzenoid B is benzene which contains a perfect matching. Assume the statement holds for a benzenoid with $k - 1 \geq 1$ hexagons, where $k \in \mathbb{N}$. Let B be a benzenoid with $h(B) = k$ hexagons. By Lemma 1 there exists a removable hexagon H in B . Let $B' = B - H$ be the catacondensed benzenoid obtained by removing H from B . Then B' contains $h(B') = h(B) - 1 = k - 1$ hexagons. By the inductive hypothesis B' contains a perfect matching. Let $M_p(B')$ be a perfect matching of B' .

By construction of catacondensed benzenoids, removing H from B removes 4 vertices from $V(B)$ and 5 edges from $E(B)$. Let $a, b, c, d \in V(B)$ be the 4 vertices in B that are not in $V(B')$. Further, let v and w be the 2 vertices from H that are elements of both $V(B')$ and $V(B)$. Then $vw \in E(B)$ and $vw \in E(B')$. Without loss of generality, suppose va, ab, bc, cd , and dw are the edges eliminated by the removal of H from B .

Attach H to B' to form B . There are two cases to consider on edge vw : either $vw \in M_p(B')$ or $vw \notin M_p(B')$.

Case 1. First suppose that edge vw is in perfect matching $M_p(B')$. Then vertices $v, w \in V(B)$ are matched by edge vw . Define $M(B) := M_p(B') \cup \{ab, cd\}$ to be a matching of B . By assumption, $M_p(B')$ is a perfect matching of B' and $V(B) \cap V(B') = V(B') - \{a, b, c, d\}$. Note that ab and cd are edges independent to edge vw that match vertices a, b, c , and d . Then B contains a perfect matching defined by $M(B) = M_p(B') \cup \{ab, cd\}$.

Case 2. Now suppose that edge vw is an element of perfect matching $M_p(B')$. Since $M_p(B')$ is a perfect matching v and w are matching by two independent edges. Define $M'(B) := M_p(B') \cup \{ab, cd\}$ to be a matching of B . Note that $V(B) \cap V(B') =$

$V(B') - \{a, b, c, d\}$. Since all vertices in B' are matched by $M_p(B')$ and ab and cd are two independent edges, $M'(B) = M_p(B') \cup \{ab, cd\}$ is a perfect matching of B .

In both cases we have shown B contains a perfect matching. It follows by induction that if benzenoid B is catacondensed, then B contains a perfect matching. \square

Conjecture 27.

$$(\sim (\text{contains_perfect_matching})) \rightarrow (\text{contains_internal_vertices})$$

Proposition 25. *Let B be a benzenoid. Suppose B does not contain a perfect matching. Then B contains internal vertices.*

Proof. Consider the contrapositive of the statement. Let B be a benzenoid that does not contain internal vertices. Then B is catacondensed. As a result of Proposition 24, benzenoid B contains a perfect matching. \square

2.6 Open conjectures

We close the investigation by stating a couple of property conjectures generated by CONJECTURING that remain open for proof.

Conjecture 28.

$$\begin{aligned} & ((\text{equal_internal_and_external_vertices}) \& (\text{even_number_of_edges})) \\ & \rightarrow (\text{even_domination_number}) \end{aligned}$$

Proposition. *Let B be a benzenoid with $n_i(B)$ internal vertices, $n_e(B)$ external vertices, $m(B)$ edges, and domination number $\gamma(B)$. If $n_i(B) = n_e(B)$ and $m(B)$ is even, then $\gamma(B)$ is even.*

Conjecture 29.

(equal_internal_and_external_vertices) \rightarrow (contains_bays)

Proposition. *Let B be a benzenoid with $n_i(B)$ internal vertices and $n_e(B)$ external vertices. If $n_i(B) = n_e(B)$, then B contains at least 1 bay.*

Bibliography

- [1] A. Bradford. *Automated Conjecturing Approach to the Discrete Riemann Hypothesis*, Master's Thesis, Virginia Commonwealth University, 2016, <http://scholarscompass.vcu.edu/etd/4470>.
- [2] G. Chartrand, and L. Lesniak. *Graphs and Digraphs*, Fifth edition; Chapman and Hall/CRC, U.S.A., 2010.
- [3] S. J. Cyvin, I. Gutman. *Topological Properties of Benzenoid Hydrocarbons*. Journal of Molecular Structure (Theochem), **150** (1987) 157-169.
- [4] M. R. Garey and D. S. Johnson. *Computers and Intractability: A Guide to the Theory of NP-Completeness*, W. H. Freeman, 1979.
- [5] I. Gutman, S. Cyvin. *Introduction to the Theory of Benzenoid Hydrocarbons*. Springer-Verlag, 1989.
- [6] T. Kikuno, N. Yoshida, and Y. Kakuda. *The NP-completeness of the dominating set problem in cubic planar graphs*. Transactions of the Institute of Electronics and Communication Engineers of Japan, E63(6):443–444, 1980.
- [7] C. E. Larson, et. al. *Domination Theory in Benzenoids* (forthcoming).
- [8] C. E. Larson and N. Van Cleemput. *Automated Conjecturing I: Fajtlowicz's Dalmatian Heuristic Revisited*, Artificial Intelligence, **231** (2016) 17-38.

- [9] M. Liedloff. *Finding a dominating set on bipartite graphs*, Information Processing Letters, **107** (2008) 154-157.
- [10] H. C. Longuet-Higgins. J. Chem. Phys., **18** (1950) 265.
- [11] L. Lovász and M. D. Plummer. *Matching Theory*, Akadémiai Kiadó - North Holland, Budapest, 1986.
- [12] Müller and I. Müller-Rodloff. Justus Liebigs Ann. Chem., **517** (1935) 134.

Appendix A

Invariant and property code

The following invariant and property-defined functions are listed in order of appearance, starting in Section 2.1.

Invariant 1 (Number of vertices).

```
return B.order()
```

Invariant 2 (Number of edges).

```
return B.size()
```

Invariant 3 (Domination number).

```
def domination_number(B):  
    return len(B.dominating_set())
```

Invariant 4 (Number of degree-2 vertices).

```
def degree_2_vertices(B):  
    return B.degree().count(2)
```

Invariant 5 (Number of degree-3 vertices).

```
def degree_3_vertices(B):  
    return B.degree().count(3)
```

Invariant 6 (Number of 2-2 edges).

```
def number_of_2_2_edges(B):  
    count = 0  
    for e in B.edges():  
        endpoint1 = e[0]  
        endpoint2 = e[1]  
        degree1 = len(B.neighbors(endpoint1))  
        degree2 = len(B.neighbors(endpoint2))  
        if degree1 == 2 and degree2 == 2:  
            count += 1  
    return count
```

Invariant 7 (Number of 2-3 edges).

```
def number_of_2_3_edges(B):  
    count = 0  
    for e in B.edges():  
        endpoint1 = e[0]  
        endpoint2 = e[1]  
        degree1 = len(B.neighbors(endpoint1))  
        degree2 = len(B.neighbors(endpoint2))  
        if (degree1 == 2 and degree2 == 3) or (degree1 == 3 and degree2 == 2):  
            count += 1  
    return count
```

Invariant 8 (Number of 3-3 edges).

```
def number_of_3_3_edges(B):
    count = 0
    for e in B.edges():
        endpoint1 = e[0]
        endpoint2 = e[1]
        degree1 = len(B.neighbors(endpoint1))
        degree2 = len(B.neighbors(endpoint2))
        if degree1 == 3 and degree2 == 3:
            count += 1
    return count
```

Invariant 9 (Matching number).

```
def matching_number(B):
    return len(B.matching())
```

Invariant 10 (Cardinality of the smaller bipartite set).

```
def smaller_bipartite_set(B):
    Sets = B.bipartite_sets()
    return min(len(Sets[0]), len(Sets[1]))
```

Invariant 11 (Cardinality of the larger bipartite set).

```
def larger_bipartite_set(B):
    Sets = B.bipartite_sets()
    return max(len(Sets[0]), len(Sets[1]))
```

Invariant 12 (Color excess).

```
def color_excess(B):  
    return larger_bipartite_set(B) - smaller_bipartite_set(B)
```

Invariant 13 (Number of hexagons).

```
def hexagons(B):  
    return B.size() - B.order() + 1
```

Invariant 14 (Number of internal vertices).

```
def internal_vertices(B):  
    return 4*hexagons(B) + 2 - B.order()
```

Invariant 15 (Number of external vertices).

```
def external_vertices(B):  
    return B.order() - internal_vertices(B)
```

Invariant 16 (Number of bay regions).

```
def number_of_bay_regions(B):  
    return number_of_2_2_edges(B) - 6
```

Invariant 17 (Number of fissures).

```
def fissures(B):  
    degrees_outer_face = [B.degree(v) for v in largest_face(B)]  
    degrees_outer_face = degrees_outer_face + degrees_outer_face[:2]  
    return len([i for i in range(len(degrees_outer_face)-2) if degrees_outer_face[  
        i:i+3]==[2,3,2]])
```

Invariant 18 (Number of bays).

```
def bays(B):
    degrees_outer_face = [B.degree(v) for v in largest_face(B)]
    degrees_outer_face = degrees_outer_face + degrees_outer_face[:3]
    return len([i for i in range(len(degrees_outer_face)-3) if degrees_outer_face[
        i:i+4]==[2,3,3,2]])
```

Invariant 19 (Number of coves).

```
def coves(B):
    degrees_outer_face = [B.degree(v) for v in largest_face(B)]
    degrees_outer_face = degrees_outer_face + degrees_outer_face[:4]
    return len([i for i in range(len(degrees_outer_face)-4) if degrees_outer_face[
        i:i+5]==[2,3,3,3,2]])
```

Invariant 20 (Number of fjords).

```
def fjords(B):
    degrees_outer_face = [B.degree(v) for v in largest_face(B)]
    degrees_outer_face = degrees_outer_face + degrees_outer_face[:5]
    return len([i for i in range(len(degrees_outer_face)-5) if degrees_outer_face[
        i:i+6]==[2,3,3,3,3,2]])
```

Invariant 21 (Number of external 2-3 edges).

```
def number_of_external_2_3_edges(B):
    return 4*hexagons(B) - 4 - 2*number_of_bay_regions(B) - 2*
        number_of_internal_vertices(B)
```

Invariant 22 (Number of external 3-3 edges).

```
def number_of_external_3_3_edges(B):  
    return B.size() - number_of_internal_3_3_edges(B) - number_of_2_2_edges(B) -  
           number_of_2_3_edges(B)
```

Invariant 23 (Number of internal 3-3 edges).

```
def number_of_internal_3_3_edges(B):  
    return hexagons(B) - 1 + internal_vertices(B)
```

Invariant 24 (Number of hexagons with 0 external edges).

```
def faces_with_0_external_edges(B):  
    count = 0  
    lf = largest_face(B)  
    leng = len(lf)  
    for f in B.faces():  
        temp_count = 0  
        if len(f) == 6:  
            for e in f:  
                x = e[0]  
                y = e[1]  
                if x in lf:  
                    x_index = lf.index(x)  
                    if x_index == 0 and (y == lf[1] or y == lf[leng-1]):  
                        temp_count +=1  
                    if x_index == leng-1 and (y == lf[0] or y == lf[leng-2]):  
                        temp_count +=1  
                    if x_index != 0 and x_index != leng-1 and (y == lf[x_index-1]  
                       or y == lf[x_index+1]):
```

```

        temp_count +=1
    if temp_count == 0:
        count +=1
return count

```

Invariant 25 (Number of hexagons with 1 external edge).

```

def faces_with_1_external_edge(B):
    count = 0
    lf = largest_face(B)
    leng = len(lf)
    for f in B.faces():
        temp_count = 0
        if len(f) == 6:
            for e in f:
                x = e[0]
                y = e[1]
                if x in lf:
                    x_index = lf.index(x)
                    if x_index == 0 and (y == lf[1] or y == lf[leng-1]):
                        temp_count +=1
                    if x_index == leng-1 and (y == lf[0] or y == lf[leng-2]):
                        temp_count +=1
                    if x_index != 0 and x_index != leng-1 and (y == lf[x_index-1]
                        or y == lf[x_index+1]):
                        temp_count +=1
            if temp_count == 1:
                count +=1
    return count

```

Invariant 26 (Number of hexagons with 2 external edges).

```
def faces_with_2_external_edges(B):
    count = 0
    lf = largest_face(B)
    leng = len(lf)
    for f in B.faces():
        temp_count = 0
        if len(f) == 6:
            for e in f:
                x = e[0]
                y = e[1]
                if x in lf:
                    x_index = lf.index(x)
                    if x_index == 0 and (y == lf[1] or y == lf[leng-1]):
                        temp_count +=1
                    if x_index == leng-1 and (y == lf[0] or y == lf[leng-2]):
                        temp_count +=1
                    if x_index != 0 and x_index != leng-1 and (y == lf[x_index-1]
                        or y == lf[x_index+1]):
                        temp_count +=1
            if temp_count == 2:
                count +=1
    return count
```

Invariant 27 (Number of hexagons with 3 external edges).

```
def faces_with_3_external_edges(B):
    count = 0
    lf = largest_face(B)
```

```

leng = len(lf)
for f in B.faces():
    temp_count = 0
    if len(f) == 6:
        for e in f:
            x = e[0]
            y = e[1]
            if x in lf:
                x_index = lf.index(x)
                if x_index == 0 and (y == lf[1] or y == lf[leng-1]):
                    temp_count +=1
                if x_index == leng-1 and (y == lf[0] or y == lf[leng-2]):
                    temp_count +=1
                if x_index != 0 and x_index != leng-1 and (y == lf[x_index-1]
                    or y == lf[x_index+1]):
                    temp_count +=1
            if temp_count == 3:
                count +=1
return count

```

Invariant 28 (Number of hexagons with 4 external edges).

```

def faces_with_4_external_edges(B):
    count = 0
    lf = largest_face(B)
    leng = len(lf)
    for f in B.faces():
        temp_count = 0
        if len(f) == 6:
            for e in f:

```

```

x = e[0]
y = e[1]
if x in lf:
    x_index = lf.index(x)
    if x_index == 0 and (y == lf[1] or y == lf[leng-1]):
        temp_count +=1
    if x_index == leng-1 and (y == lf[0] or y == lf[leng-2]):
        temp_count +=1
    if x_index != 0 and x_index != leng-1 and (y == lf[x_index-1]
        or y == lf[x_index+1]):
        temp_count +=1
if temp_count == 4:
    count +=1

return count

```

Invariant 29 (Number of hexagons with 5 external edges).

```

def faces_with_5_external_edges(B):
    count = 0
    lf = largest_face(B)
    leng = len(lf)
    for f in B.faces():
        temp_count = 0
        if len(f) == 6:
            for e in f:
                x = e[0]
                y = e[1]
                if x in lf:
                    x_index = lf.index(x)
                    if x_index == 0 and (y == lf[1] or y == lf[leng-1]):

```

```

        temp_count +=1
    if x_index == leng-1 and (y == lf[0] or y == lf[leng-2]):
        temp_count +=1
    if x_index != 0 and x_index != leng-1 and (y == lf[x_index-1]
        or y == lf[x_index+1]):
        temp_count +=1
    if temp_count == 5:
        count +=1
return count

```

Invariant 30 (Number of leaves).

```

def leaves(B):
    return inner_dual_benzenoid(B).degree().count(1)

```

Invariant 31 (Branching number).

```

def branching_number(B):
    return inner_dual_benzenoid(B).degree().count(3) + count_cyclic_sublists([B.
        degree(v) for v in largest_face(B)], [3,2,2,3])

```

Property 1 (Even number of vertices).

```

def even_number_of_vertices(B):
    return B.order()%2==0

```

Property 2 (Odd number of vertices).

```

def odd_number_of_vertices(B):
    return B.order()%2==1

```

Property 3 (Even number of edges).

```
def even_number_of_edges(B):  
    return B.size()%2==0
```

Property 4 (Odd number of edges).

```
def odd_number_of_edges(B):  
    return B.size()%2==1
```

Property 5 (More edges than vertices).

```
def more_edges_than_vertices(B):  
    return (B.size() > B.order())
```

Property 6 (Odd number of hexagons).

```
def odd_hexagons(B):  
    return hexagons(B)%2==1
```

Property 7 (Even number of hexagons).

```
def even_hexagons(B):  
    return hexagons(B)%2==0
```

Property 8 (Catacondensed).

```
def catacondensed(B):  
    return internal_vertices(B)/external_vertices(B)==0
```

Property 9 (Even domination number).

```
def even_domination_number(B):  
    return domination_number(B)%2==0
```

Property 10 (Odd domination number).

```
def odd_domination_number(B):  
    return domination_number(B)%2==1
```

Property 11 (Equality).

```
def equality(B):  
    return hexagons(B) == domination_number(B)
```

Property 12 (Equal internal and external vertices).

```
def equal_internal_and_external_vertices(B):  
    return internal_vertices(B)/external_vertices(B)==1
```

Property 13 (Contains internal vertices).

```
def contains_internal_vertices(B):  
    return internal_vertices(B)!=0
```

Property 14 (Balanced).

```
def balanced(B):  
    return smaller_bipartite_set(B) == larger_bipartite_set(B)
```

Property 15 (Contains color excess).

```
def contains_color_excess(B):  
    return color_excess(B) != 0
```

Property 16 (Contains bay region).

```
def contains_bay_region(B):  
    return number_of_bay_regions(B) > 0
```

Property 17 (Contains fissures).

```
def contains_fissures(B):  
    return fissures(B) != 0
```

Property 18 (Contains bays).

```
def contains_bays(B):  
    return bays(B) != 0
```

Property 19 (Contains coves).

```
def contains_coves(B):  
    return coves(B) != 0
```

Property 20 (Contains fjords).

```
def contains_fjords(B):  
    return fjords(B) != 0
```

Property 21 (Contains hexagon with 0 external edges).

```
def contains_face_with_0_external_edges(B):  
    return faces_with_0_external_edges(B) != 0
```

Property 22 (Contains hexagon with 1 external edge).

```
def contains_face_with_1_external_edge(B):  
    return faces_with_1_external_edge(B) != 0
```

Property 23 (Contains hexagon with 2 external edges).

```
def contains_face_with_2_external_edges(B):  
    return faces_with_2_external_edges(B) != 0
```

Property 24 (Contains hexagon with 3 external edges).

```
def contains_face_with_3_external_edges(B):  
    return faces_with_3_external_edges(B) != 0
```

Property 25 (Contains hexagon with 4 external edges).

```
def contains_face_with_4_external_edges(B):  
    return faces_with_4_external_edges(B) != 0
```

Property 26 (Contains hexagon with 5 external edges).

```
def contains_face_with_5_external_edges(B):  
    return faces_with_5_external_edges(B) != 0
```

Property 27 (Contains perfect matching).

```
def contains_perfect_matching(B):  
    nu = B.matching(value_only=True)  
    return nu == B.order()/2
```
