How Constraint Programming Can Help Chemists to Generate Benzenoid Structures and Assess the Local Aromaticity of Benzenoids *†

Yannick Carissan[1] Denis Hagebaum-Reignier[1]
Nicolas Prcovic[2] Cyril Terrioux[2] Adrien Varet[2]
[1] Aix Marseille Univ, CNRS, Centrale Marseille, ISM2, Marseille, France
[2] Aix Marseille Univ, Université de Toulon, CNRS, LIS, Marseille, France
{firstname.name}@univ-amu.fr

Abstract

Benzenoids are a subfamily of hydrocarbons (molecules that are only made of hydrogen and carbon atoms) whose carbon atoms form hexagons. These molecules are widely studied in theoretical chemistry and have a lot of concrete applications. Then, there is a lot of problems relative to this subject, like the enumeration of all its Kekulé structures (i.e. all valid configurations of double bonds). In this article, we focus our attention on two issues: the generation of benzenoid structures and the assessment of the local aromaticity.

On the one hand, generating benzenoids that have certain structural and/or chemical properties (e.g. having a given number of hexagons or a particular structure from a graph viewpoint) is an interesting and important problem. It constitutes a preliminary step for studying their chemical properties. In this paper, we show that modeling this problem in Choco Solver and just letting its search engine generate the solutions is a fast enough and very flexible approach. It can allow to generate many different kinds of benzenoids with predefined structural properties by posting new constraints, saving the efforts of developing bespoke algorithmic methods for each kind of benzenoids.

On the other hand, we want to assess the local aromaticity of a given benzenoid. This is a central issue in theoretical chemistry since aromaticity cannot be measured. Nowadays, computing aromaticity requires quantum chemistry calculations that are too expensive to be used on medium to large-sized molecules. In this article, we describe how constraint programming can be useful in order to assess the aromaticity of benzenoids. Moreover, we show that our method is much faster than the reference one, namely NICS.

Keywords: Constraint programming, Graph variables and constraints, Modeling, Theoretical chemistry

1 Introduction

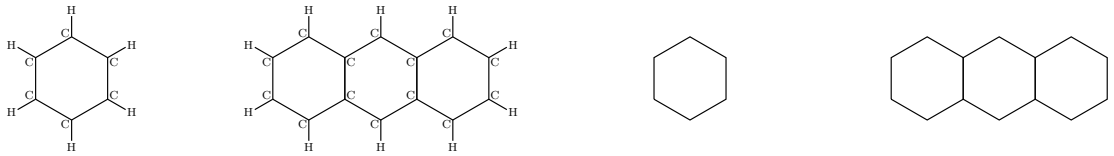
Polycyclic aromatic hydrocarbons (PAHs) are hydrocarbons whose carbon atoms are forming cycles of different sizes. *Benzenoids* are a subfamily of PAHs made of 6-membered carbon rings (i.e. each cycle is a hexagon). To fill carbon valency, each atom of carbon is bonded to either two other carbons and one hydrogen or three carbons. For example, Figures 1(a) and (b) are representing two benzenoids: benzene and anthracene.

PAHs are well-studied in various fields because of their energetic stability, molecular structures or optical spectra. In a natural environment, these molecules are created by the incomplete combustion of carbon contained in combustibles [Luch, 2005]. They are popular research subjects in material sciences, e.g. molecular nanoelectronics where they are used to store or transport energy [Wu et al., 2007, Aumaitre and Morin, 2019] or in organic synthesis [Rieger and Müllen, 2010, Narita et al., 2015], where the controlled design of specific shapes remains challenging. PAHs are also intensively studied in interstellar chemistry because of their suspected presence in various interstellar and circumstellar environments where they are believed to act as catalysts for chemical reactions taking place in space [Draine, 2011]. They are also intensively studied in other domains like molecular nanoelectronics [Wu et al., 2007].

In this context, this article addresses two important issues in theoretical chemistry. The first one concerns the generation of specific benzenoid structures while the second one considers the assessment of the local aromaticity thanks to the computation of the local resonance energy.

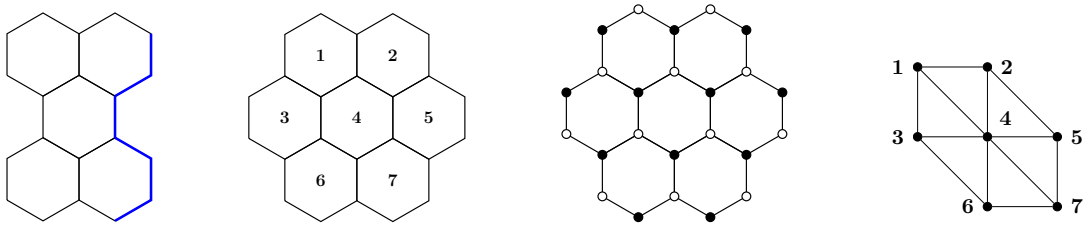
*This work has been funded by the Agence Nationale de la Recherche project ANR-16-CE40-0028.

†This version of the article has been accepted for publication, after peer review but is not the Version of Record and does not reflect post-acceptance improvements, or any corrections. The Version of Record is available online at: http://dx.doi.org/10.1007/s10601-022-09328-x. Use of this Accepted Version is subject to the publisher's Accepted Manuscript terms of use https://www.springernature.com/gp/open-research/policies/accepted-manuscript-terms.



(a) (b) (c) (d)

Figure 1: Two small benzenoids: benzene (a) and anthracene (b) with their graphical representations (c) and (d).



(a) (b) (c) (d)

Figure 2: Definition of armchair edge (depicted by thick blue lines) of benzenoids (a), coronene (b), a bipartition of its carbon atoms (c) and its hexagon graph (d).

Regarding the first issue, PAHs are well-known to exhibit a large variety of physicochemical properties depending on size and, more specifically, on edge and bond topologies. In the astrophysical community, the so-called "PAH hypothesis" formulated more than 30 years ago, of whether the existence of PAHs in space could explain some unidentified mid-infrared emission bands in astrophysical environments, has motivated numerous observational, experimental and theoretical investigations. It is now accepted that mixtures of free PAHs of different sizes, shapes and ionic states can account for the overall appearance of the widespread interstellar infrared emission spectrum. But the question of relative abundance of PAHs with a given size and/or shape remains open. Many studies are devoted to exploring the effect of the size, shapes in terms of compacity and symmetry of PAHs, on band positions and intensities of the infrared spectra [Allamandola et al., 1999, Bauschlicher et al., 2008, Ricca et al., 2012]. Very recently, a systematic investigation of a series of 328 PAHs containing up to 150 carbon atoms showed that PAHs with armchair edges (an armchair edge is a particular configuration of the edge of the PAH as we can see in Figure 2(a) that maximize the Clar number (i.e. the maximum number of non-adjacent rings containing 6 electrons called a sextet) are potential emitters of a certain class of astrophysical infrared sources [Ricca et al., 2019]. For their study, the authors needed to systematically generate all PAHs having armchair edge topology and selecting a subclass of PAHs whose structure maximizes the Clar number. They used the algorithm of Caporossi and Hansen [Caporossi and Hansen, 1998]. Constraint Programming (CP) is particularly well suited for the generation of such families of PAHs.

Another interesting example where the generation of specific shapes is relevant for chemists deals with so-called "non-Kekulean" benzenoids [Cyvin et al., 1989]. These benzenoids cannot be represented by Kekulé structures, i.e. structures that have only simple and double bonds. From a graph-theoretical point of view, Kekulé structures are covered by the maximal number of disjoint (double) edges so that all vertices are incident to one of the disjoint edges. Until recently, chemists commonly admitted that "non-Kekulean" benzenoids were very unstable due to their open-shell electronic structure (i.e. one or more electron(s) remain unpaired, contrary to a closed-shell structure where all electrons are paired) and thus that their synthesis would be a real challenge. The experimental realization and in-depth characterization of small "non-Kekulean" benzenoids were very recently achieved on gold surfaces [Mishra et al., 2019, 2020]. These studies paved the way for the synthesis of new classes of compounds that show unconventional magnetism induced by their topology, with promising applications in various fields like molecular electronics, nonlinear optics, photovoltaics and spintronics. Moreover, it was shown that some PAHs with specific topologies (e.g. rhombus shapes) may "prefer" having an open-shell structure when reaching a certain size, although they could have a closed-shell structure and could thus be described by a set of Kekulé structures [Trinquier and Malrieu, 2018]. Theoretically, from a quantum point of view, the proper description of the electronic structure of open-shell benzenoids is a difficult task. The use of a constraint programming approach for the systematic search of larger non-Kekulean or Kekulean benzenoids having an open-shell electronic structure is undoubtedly advantageous.

In this context, many approaches have been proposed in order to generate benzenoids possibly having a particular shape or satisfying a particular property (e.g. [Brinkmann et al., 2002, Brunvoll et al., 1990]). These are bespoke approaches which have the advantage of being efficient but which are difficult to adapt to the needs of chemists. Moreover, designing a new bespoke method for each new desired property often requires a huge amount of effort. So, in this paper, we prefer to use a more generic approach based on constraint programming. With this aim in view, we present a general model

which can be refined depending on the desired properties by simply adding variables and/or constraints. By doing so, our approach benefits from the flexibility of CP and requires fewer efforts of implementation. In the meantime, CP offers efficient solvers which can be quite competitive with respect to bespoke algorithms.

The second issue we consider deals with the *aromaticity* of benzenoids, which is a fundamental concept in chemistry (defined in Section 2). Since the discovery of graphene by Andre Geim and Konstantin Novoselov awarded with the 2010 Nobel price in physics, the interest in aromaticity vividly revived due to its potential importance in nanoelectronics. Indeed, aromaticity favors electronic flow through molecules and, thus, aromatic compounds are of interest for the design of nanoelectronic compounds. This concept allows chemists to link the energetic stability of a molecule to its molecular structure [Clar and Schoental, 1964]. The stability of a molecule is a measure of the energy needed to break all chemical bonds and separate all the atoms of the molecule apart. Because of aromaticity, some molecules have an extra term in this energy: breaking them apart requires more energy than for non-aromatic molecules with the same number of atoms. Recently, some methods using quantum chemistry were established in order to assess the aromaticity of a given molecule. The most popular one called *NICS* (Nuclear Independent Chemical Shift [Chen et al., 2005]) consists of applying a magnetic field perpendicular to the molecular plane and observe the response of the electronic cloud, which gives insight into electronic currents. NICS values are indicators of the intensity of these currents and are complementary to energy based methods to describe aromaticity. Analyzing the response of the electronic density allows chemists to quantify the aromaticity of the molecule. However, this method has a very high cost and computing the aromaticity of large molecules can easily take a few days. This large computational cost is due to the fact that quantum chemistry calculations require many steps involving iterative procedures before doing the actual calculation of aromaticity. To circumvent this drawback, some methods using graph theory were proposed in the 1990s [Randić et al., 1996, Lin and Fan, 1999, Lin, 2000], which roots can be tracked back to the work of Hückel in the 1930s [Hückel, 1931]. They will be presented in the following parts.

In this paper, we describe how constraint programming can help chemists to compute the local aromaticity of benzenoids. More precisely, we show that some tasks can easily be modeled as CSP instances and solved efficiently thanks to constraint solvers like Choco [Fages et al.] while requiring a reduced implementation effort unlike usual methods from theoretical chemistry or any bespoke methods based on algorithm engineering. Among these tasks, we can cite the enumeration of particular cycles or one of Kekulé structures.

The paper is an extension of previous work published in [Carissan et al., 2020a,b]. It differs notably by improving some models (e.g. the general model or the coronenoid model) and by taking into account new properties for structure generation (e.g. having a given number of carbon/hydrogen atoms or a given irregularity) while we consider an additional method for assessing aromaticity. It is organized as follows. First, we recall some definitions about benzenoids and constraint programming in Section 2. Section 3 introduces the fastest existing algorithm to our knowledge for generating benzenoid structures while Section 4 presents our approach using constraint programming, explains its advantages and illustrates its potential by giving some examples. Section 5 is devoted to an empirical evaluation of some of the models we propose. Regarding the second issue, Section 6 introduces some existing methods to assess the aromaticity of benzenoids thanks to the local resonance energy. In Section 7, we describe two new methods which exploit constraint programming in order to compute the local resonance energy. In Section 8, we present some experimental results which show the interest of our approach. Finally, we conclude and provide some perspectives in Section 9.

2 Preliminaries

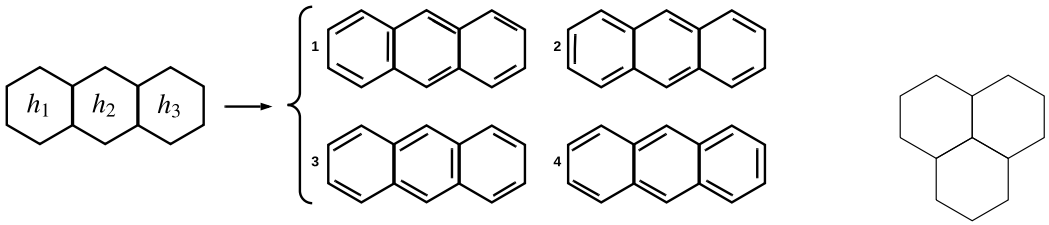
2.1 Theoretical Chemistry

Benzene, represented in Figure 1(a) is a molecule made of 6 carbon atoms and 6 hydrogen atoms. Its carbon atoms form a hexagon (also called *benzenic cycle* or *benzenic ring*) and each of them is linked to a hydrogen atom. *Benzenoids* are a subfamily of PAHs containing all molecules which can be obtained by aggregating (or fusing) benzenic rings. For example, Figure 1(b) shows anthracene, which contains three benzenic rings.

Hereafter, we first define some graphs related to benzenoids and then we recall the concept of aromaticity.

2.1.1 Benzenoids and Graphs

In a benzenoid, each carbon atom is linked either to two other carbon atoms and one hydrogen atom or three other carbon atoms. Therefore, benzenoids can be perfectly defined by describing only the interactions between carbon atoms. Hydrogen atoms can then be deduced since each hydrogen atom is linked to a carbon atom which is only bonded to two other carbon atoms. As such, any benzenoid can be represented as an undirected graph $B = (V, E)$, with V the set of vertices and E the set of edges. Every vertex in V represents a carbon atom and every edge of E represents a bond between the two corresponding carbons. Moreover, this kind of graph is connected, planar and bipartite. Figures 1(c) and (d) represent the graphs related to the molecules of benzene and anthracene. In Figure 2(c), we show why the graph related to coronene (a well-known benzenoid depicted in Figure 2(b)) is bipartite by giving the two disjoint and independent sets. These sets are defined by the white and black vertices respectively.



(a) (b)

Figure 3: Kekulé structures of anthracene (a). A benzenoid having no Kekulé structure (b).

When dealing with the generation of benzenoid structures, for any benzenoid B, we need to consider some of its faces. A *face* of a planar graph is an area of the plan bounded by edges. For instance, the graph in Figure 2(b) has eight faces namely, the seven numbered faces and the external face. Note that, in most cases, we do not take into account the external face and so the considered faces correspond exactly to the hexagons. For example, this holds for coronene as we can see in Figure 2(b). However, we will see later that the faces and hexagons may not match when the benzenoid contains at least one hole (see Subsection 4.4 for more details).

In addition, given a benzenoid, we consider another graph, namely the hexagon graph. The *hexagon graph* of a benzenoid $B = (V, E)$ is the undirected graph $B_h = (V_h, E_h)$ such that there is a vertex v_h from V_h per hexagonal face h of B (the external face and "holes" in the benzenoid are excluded) while there is an edge $\{v_h, v_{h'}\}$ in E_h if the corresponding hexagonal faces h and h' of B share an edge of E. Figure 2(d) presents the hexagon graph of coronene. The hexagon graph allows us to express the interaction between the hexagons of the considered benzenoid.

2.1.2 Aromaticity

From a chemical viewpoint, the *valence* of an atom is the number of bonds that it can establish with its electrons (one electron per bond). Carbon and hydrogen atoms have a valence of 4 and 1 respectively. So, we can easily deduce that in benzenoids, each carbon atom has one of its electrons that is not used. These electrons are called *π-electron* and can be used to enhance one bond by establishing *double bonds* (i.e. a bond involving two electrons per atom). Therefore, in benzenoids, each carbon is involved in a double bond and two single bonds.

A *Kekulé structure* of a benzenoid is a valid configuration of its double bonds (i.e. a configuration in which each carbon atom is involved in exactly one double bond). Figure 3(a) depicts all the Kekulé structures of anthracene. A benzenoid can have several Kekulé structures or none (Figure 3(b) depicts an example of benzenoid which has no Kekulé structure). We denote $\mathcal{K}(B)$ the set of all Kekulé structures of a benzenoid B. Note that the number of Kekulé structures of a benzenoid can be exponential in the number of carbon atoms. Therefore, given a benzenoid, generating all its Kekulé structures is a hard problem. A benzenoid continually alternates between its Kekulé structures. This dynamic is at the origin of the notion of aromaticity. There exist some methods based on graph theory that allow computing the resonance energy of a given benzenoid (i.e. the energy induced by its aromaticity) and these methods require to be able to enumerate all its Kekulé structures [Randić, 2003].

Aromaticity is a concept built by chemists in the early 20^{th} century in order to account for the surprising chemical stability of the benzene molecule. In this molecule, after making a single bond to each of its three neighbors (two carbon and one hydrogen), each carbon of the hexagonal geometry carries one extra electron. Electrons tend to form bonds (i.e. pair with another electron) whenever possible. Thus, this electron forms a molecular bond with the electron of a neighboring carbon atom. When all six electrons do the same, the electronic structure, first proposed by Kekulé [Kekulé, 1866], is obtained. Yet, two such structures exist as the pairing for one carbon can be with any of its two neighbors. The interaction or resonance of these two coexisting solutions is described by quantum physics and leads to an over-stabilization energy called aromaticity. This concept can be extended to fused benzene rings. It turns out that aromatic molecules often have a characteristic smell and/or taste, hence the name of the concept.

Due to the physical nature of aromaticity, hydrogen atoms do not play any role in its determination. Thus, it is custom not to take them into account in connectivity-based methods. So, representing a benzenoid by a graph whose vertices correspond to carbon atoms as mentioned above is sufficient. Finally, from the viewpoint of graph theory, we can remark that the set of double bonds of a Kekulé structure is nothing more than a perfect matching on the benzenoid. As a reminder, a *perfect matching* of an undirected graph $G = (V, E)$ is a set of edges $E' \subseteq E$ such that $\forall (e_1, e_2) \in E' \times E', e_1 \neq e_2, e_1 \cap e_2 = \emptyset$ and $\bigcup_{e \in E'} e = V$.

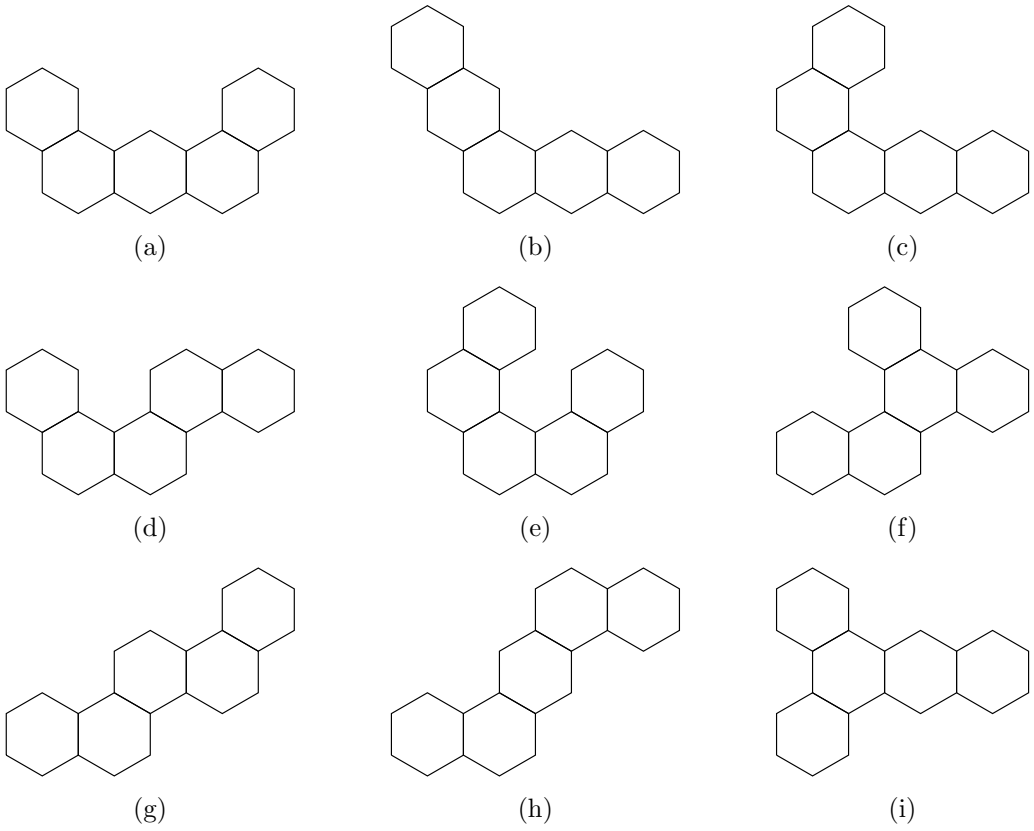


Figure 4: Catacondensed benzenoids with 5 hexagons.

2.2 Constraint Programming

An instance I of the *Constraint Satisfaction Problem (CSP)* is a triplet (X, D, C). $X = \{x_1, \ldots, x_n\}$ is a set of n *variables*. For each variable $x_i \in X$, there exists an associated domain $D_{x_i} \in D = \{D_{x_1}, \ldots, D_{x_n}\}$ which represents the values that x_i can take. $C = \{c_1, \ldots, c_e\}$ represents a set of e *constraints*. Constraints represent the interactions between the variables and describe the allowed combinations of values.

Solving a CSP instance $P = (X, D, C)$ amounts to finding an assignment of all the variables of X with a value contained in their associated domain which satisfies all the constraints of C. Such an assignment is called a *solution*. This problem is NP-hard [Rossi et al., 2006].

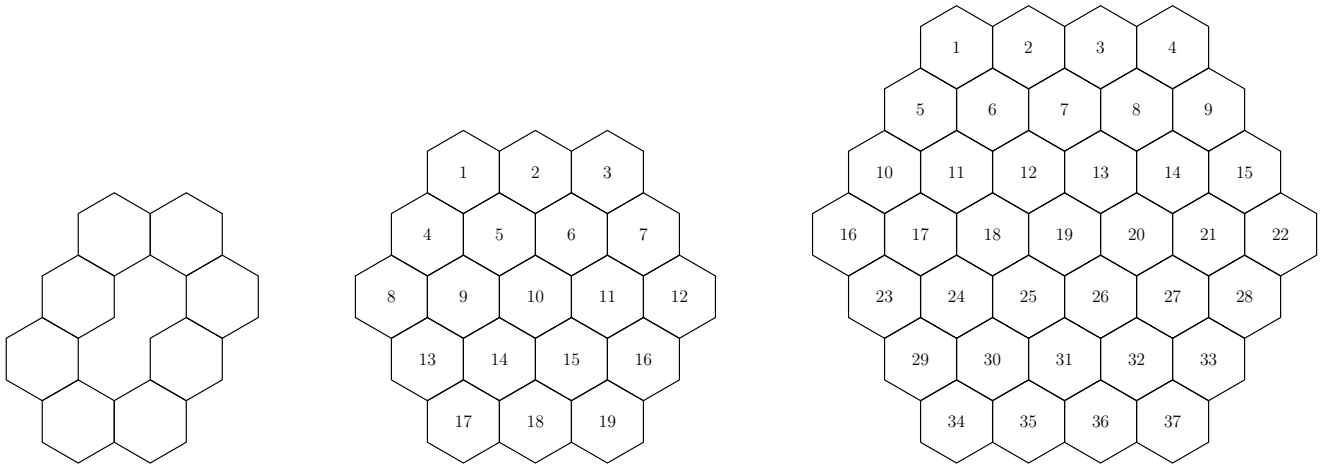
Many libraries are available to represent and efficiently solve CSP instances. In this paper, we exploit the open-source Java library *Choco* [Fages et al.]. This choice is highly guided by our need to be able to define *graph variables* and directly apply graph-related constraints (e.g. connected or cyclic constraints). Graph variables have as domain a set of graphs defined by a lower bound (a sub-graph called GLB) and an upper bound (a super-graph called GUB). Moreover, Choco implements the usual global constraints which make the modeling easier and its solver is efficient and configurable.

3 Generating Benzenoid Structures

We can define the benzenoid generation problem (denoted BGP in the future) as follows: given a set of structural properties $\mathcal{P}$, generate all the benzenoids which satisfy each property of $\mathcal{P}$. For instance, these structural properties may deal with the number of carbons, the number of hexagons or a particular structure for the hexagon graph. Naturally, the most interesting instances of the BGP problem combine several properties. For example, Figure 4 shows benzenoids having a tree as hexagon graph and possessing five hexagons. Such a property-based design of instances allows for the search of benzenoids with chemically relevant properties. Our interest lies in the search of benzenoids with radical electronic structures (as in the work of Malrieu and Trinquier [Trinquier and Malrieu, 2018]), which arise from their geometrical arrangement[1].

Next, we present an existing method proposed by Brinkmann et al. [Brinkmann et al., 2002]. Given an integer n, this method is able to generate all the benzenoids with n hexagons by generating all the hexagon graphs with at most n

[1] A radical structure arises when a system has an odd number of electrons. All electrons but one pair form bonds. The lonely electron is called a radical.



(a) (b) (c)

Figure 5: The smallest benzenoid with hole (a) and coronenoids of size 3 (b) and 4 (c).

vertices. This is done by successively adding new vertices to the hexagon graph (which is equivalent to generate all the wanted molecules by successively adding new hexagons).

This method is highly efficient. For instance, it could generate the 669,584 benzenoids having 12 hexagons in 1.2 seconds and 1,000 billion of benzenoids having 21 hexagons in two weeks when launched on an old computer (Intel Pentium, 133 MHz, 2002). However, it has some disadvantages. Indeed, it is not complete in the sense that it is unable to generate benzenoids with "holes". By hole, we mean a face that does not correspond to a hexagon or to the external face. For example, Figure 5(a) depicts the smallest benzenoid (in terms of the number of hexagons) which admits a hole. Such a benzenoid cannot be produced by this method. Indeed, when this method wants to add a new hexagon, it checks whether the added hexagon allows closing a cycle of hexagons. If so, the hexagon is not added and, therefore, benzenoids with holes cannot be generated. Benzenoids with holes are quite rare. There is a single one for 8 hexagons (among 1,436 benzenoids), 5 for 9 hexagons (among 6,510). Note that this proportion grows as we increase the number of hexagons (see Tables 3 and 5). Furthermore, this method is unable to take into account other properties natively and cannot easily be tuned to fit the needs of chemists. Indeed, it is based on an augmenting procedure that decides how to add a vertex. Consequently, this procedure should be changed and proven adequate to avoid generating non-canonical graphs (i.e. redundant isomorphic graphs) each time the structural property of the benzenoids to generate is changed. This can be a relatively heavy task even for the addition of a basic property.

In the next section, we present a new method using constraint programming which is able to generate any benzenoid structure and which benefits from the flexibility of constraint programming.

4 Generating Benzenoid Structures Thanks to CP

In this section, we see how to model a BGP instance as a CSP instance. We first present a general model which considers the generation of all the benzenoids having a given number of hexagons. This property is the minimal property to ensure. Then we provide some examples showing how the model can be easily specialized to take into account some additional structural properties.

4.1 General Model

In this part, we want to generate all the benzenoids having a given number n of hexagons. Before modeling this problem as a CSP instance, we highlight some useful properties. A *coronenoid* of size k is a benzene molecule (i.e. a hexagon) to which we successively add $k-1$ crowns of hexagons. Benzene corresponds to the coronenoid of size 1 (see Figure 1(c)). Figures 2(b) and 5(b)-(c) present the coronenoids of size 2 (called coronene), 3 and 4. Remark that the number of hexagons in the ith crown grows with i. Furthermore, note that the diameter (i.e. the number of hexagons of the central line) of a coronenoid of size k is $2 \times k - 1$. Our interest in coronenoids lies in the fact that they are useful to "embed" benzenoids of a given number of hexagons.

Property 1 *Any benzenoid involving n hexagons can be embedded in a coronenoid of size at most $k(n) = \left\lfloor \frac{n}{2} + 1 \right\rfloor$.*

If we reason in terms of hexagon graph, obtaining all the benzenoids with n hexagons is equivalent to finding all the

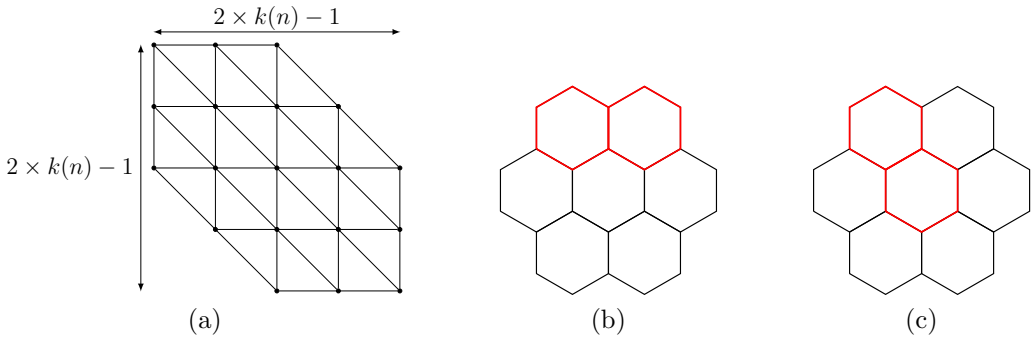


Figure 6: Upper bound of the domain of the graph variable (a) Two structures (in red solid line) of 2 hexagons embedded in a coronenoid of size 2 which are not detected as being the same benzenoid by the constraint `lex-lead` (b) and (c).

connected sub-graphs[2] of the hexagon graph of the coronenoid of size $k(n)$. Hereafter, we denote $B^{c(k(n))}$ the coronenoid of size $k(n)$ and $B_h^{c(k(n))}$ its hexagon graph. The model we propose relies on this property.

Given an integer n, we model the BGP problem where $\mathcal{P}$ is reduced to "having n hexagons" as a CSP instance $I = (X, D, C)$. First, we consider a graph variable x_G which represents the possible hexagon graph of the built benzenoid. Its domain is the set of all the sub-graphs between the empty graph and the hexagon graph of the coronenoid of size $k(n)$ (see Figure 6(a)). We also exploit a set of n_c Boolean variables $\{x_1, \ldots, x_{n_c}\}$ where n_c is the number of hexagons of the coronenoid of size $k(n)$. The variable x_i is set to 1 if the ith hexagon of the coronenoid of size $k(n)$ is used in the hexagon graph depicted by x_G, 0 otherwise. For the sake of simplicity, hexagons are numbered from top to bottom and from left to right like in Figure 2(b) or Figures 5(b)-(c). Likewise, given m_c the number of edges of the hexagon graph of the coronenoid of size $k(n)$, we consider a set of m_c Boolean variables $y_{i,j}$. The variable $y_{i,j}$ is set to 1 if the edge $\{i, j\}$ of the hexagon graph of the coronenoid of size $k(n)$ is used in the hexagon graph depicted by x_G, 0 otherwise.

Finally, we model the following properties by constraints:

- *Link between the graph variable x_G and the variables x_i* As mentioned above, the variable x_i specifies if the ith hexagon of the coronenoid of size $k(n)$ is used in the graph represented by x_G. So we must ensure that their respective values are consistent with each other. With this aim in view, we consider a channeling constraint per variable x_i which involves x_i and x_G and imposes that $x_i = 1 \iff x_G$ *contains the vertex* i.

- *Link between the graph variable x_G and the variables $y_{i,j}$* We similarly consider a channeling constraint per variable $y_{i,j}$ which involves $y_{i,j}$ and x_G and imposes that $y_{i,j} = 1 \iff x_G$ *contains the edge* $\{i, j\}$.

- *x_G is an induced sub-graph of the coronenoid hexagon graph.* Any value of x_G is not necessarily a valid hexagon graph. For example, in Figure 2(d), removing only edge $\{1, 2\}$ does not produce a valid hexagon graph. To ensure that the hexagon graph is valid, it must correspond to a sub-graph of the coronenoid hexagon graph induced by the vertices belonging to x_G. So for every edge $\{i, j\}$ in the coronenoid hexagon graph, we add a constraint $x_i = 1 \wedge x_j = 1 \Leftrightarrow y_{i,j} = 1$. In other words, the edge $\{i, j\}$ exists in x_G if and only if the vertices i and j appear in x_G.

- *Benzenoids have n hexagons* It can be easily modeled by using a sum global constraint involving all the variables x_i: $\sum_{i \in \{1,\ldots,n_c\}} x_i = n$.

- *Benzenoids correspond to connected graphs* Variable graphs come with particular constraints. Among them, we consider the `connected` constraint which applies on the variable x_G ensuring that only connected graphs are allowed values for x_G.

- *Six hexagons forming a cycle generate a hexagon* When six hexagons form a cycle, the face contained in the interior of the cycle is not a hole but a hexagon. For instance, if we consider the cycle formed by the hexagons 1, 2, 5, 7, 6 and 3 of coronene (see Figure 2(b)), we have necessarily a hexagon in the middle of the coronenoid, namely the hexagon 4. To ensure this property, we add a set of constraints specifying that G cannot have a hole whose size is exactly one hexagon. For each hexagon u, we consider the set $N(u)$ of the neighbors of u in the hexagon graph. Then, for each vertex u having 6 neighbors, we add a constraint between x_u and the variables corresponding to its neighbors which imposes: $\sum_{v \in N(u)} x_v = 6 \Rightarrow x_u = 1$.

[2]Remember that a graph (V', E') is a *sub-graph* of a graph (V, E) if $V' \subseteq V$ and $E' = E \cap (V \times V)$.

This model allows us to enumerate all the benzenoids having n hexagons, possibly with holes. However, some benzenoids may be generated multiple times due to the existence of symmetries. So we add several additional constraints in order to break as many symmetries as possible:

- Two constraints specify that G must have at least one vertex respectively on the top-border and the left-border in order to avoid the symmetries by translation. Let T (respectively L) be the set of hexagons on the top-border (resp. left-border) of the coronenoid of size $k(n)$. For instance, if we consider the coronenoid of size 3 (see Figure 5(b)), we have $T = \{1, 2, 3\}$ and $L = \{1, 4, 8, 13, 17\}$. We have to create a constraint specifying that the sum of the binary variables associated with the top border (resp. left border) is strictly positive, that is $\sum_{h \in T} x_h > 0$ (resp. $\sum_{h \in L} x_h > 0$). This can also be expressed as the clauses $\bigvee_{h \in T} x_h$ and $\bigvee_{h \in L} x_h$. In our implementation, we retain this last possibility which turns out to be more efficient in practice.
- A set of constraints specify that G must be the only representative of its class of symmetry by axis and rotation. There are up to twelve symmetric solutions: six 60 degrees rotation symmetries combined with a possible axis symmetry. Symmetries are broken thanks to the compact `lex-lead` constraint described in [Devriendt et al., 2016]. For each of the twelve symmetries, it requires n_c new Boolean variables (each associated with a Boolean variable x_i representing a hexagon) and a total of $3n_c$ ternary clauses.

Note that the latter set of constraints only allows avoiding symmetric solutions obtained by rotations with respect to the hexagon located at the center of the coronenoid. For instance, the two structures depicted in Figures 6(b)-(c) are not detected as expressing the same benzenoid by the constraint `lex-lead`. In order to filter the remaining symmetric solutions, we forbid some assignments thanks to nogoods. More precisely, every time a solution is found, we compute all the structures that can be obtained by a translation or a rotation while having a hexagon on the top-border or left-border of the coronenoid. Each of these structures can be characterized by the set S of hexagons h for which the variable x_h is assigned to true. In order to avoid generating the structure characterized by the set S, we post the nogood $\bigvee_{h \in S} \overline{x_h}$ where $\overline{x_h}$ denotes the negative literal related to x_h. Remark that, by doing so, the `lex-lead` constraint becomes redundant. However, the clauses it produces are shorter than ones related to nogoods and so may be exploited earlier to prune the search tree. So its preservation in the model allows making the solving more efficient.

This model can be easily implemented with the open-source Java library *Choco* [Fages et al.]. Indeed, Choco natively proposes graph variables and the more usual graph-related constraints (notably `connected` constraint).

As our model mainly exploits clauses, we can ask ourselves the question of formulating it directly in the form of an SAT instance. The main difficulty in proposing a SAT model lies in the representation of the hexagon graph and its properties. For instance, expressing a connected graph is far from being an easy task in SAT. Moreover, remember that the graph is not known at the beginning but computed during the solving. So using graph variables and their related constraints is much simpler.

4.2 How to Specialize the Model

The first advantage of our approach is that it is able to generate all the benzenoids, including those with holes unlike the method described in the previous section. Moreover, using constraint programming makes the addition of most of the structural properties wished by the chemists easier. Indeed, starting from the general model, for each new desired property, we simply have to model it by posting new constraints and possibly by adding new variables.

For example, if chemists are interested in benzenoids whose structure is a path of hexagons, they can be easily generated by exploiting the general model I and adding the graph constraint `path` on x_G. Similarly, if chemists are more interested in *catacondensed benzenoids*, that is benzenoids whose structure is a tree, we can just add the graph constraint `tree` on x_G to the general model I. Figure 4 shows nine (among twelve possible) examples of 5-hexagon benzenoids obtained by just adding the `tree` constraint of Choco on x_G.

Clearly, depending on the desired property, the model may be more complex. It may specifically require adding new variables if the property cannot be directly expressed by some existing constraints. In the next subsections, we give such examples.

4.3 Generating Benzenoid Structures Having a Rectangular or Rhombic Shape

In this part, we present how we can model the property *"all the built benzenoids have a rectangle shape"*, in addition to the property *"having n hexagons"*. Let us specify that the rectangular forms which interest the chemists are full rectangles [Rayne and Forest, 2011, Dias, 2013]. For instance, Figure 8 shows all the rectangle benzenoid structures with at most six hexagons.

First, recall that the general model described in the previous part takes as input the number n of hexagons, and embeds any generated benzenoid in a coronenoid of size $k(n)$. We can easily see that the largest rectangle benzenoid which can

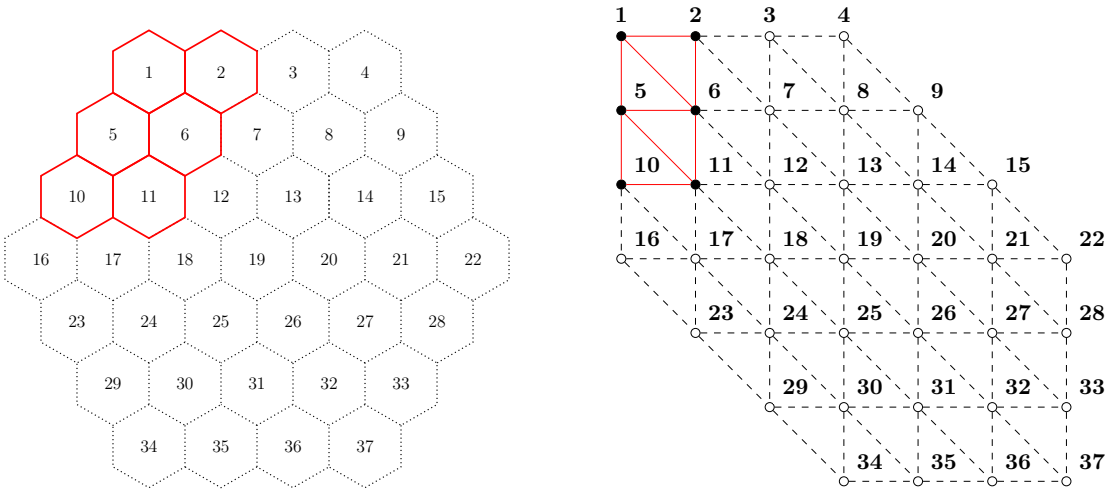


Figure 7: Rectangle benzenoid (in red solid line) of dimension 2×3 embedded in the coronenoid of size 4 (a) and its related hexagon graph (b).

be embedded in a coronenoid of size $k(n)$ has a width w_{max} equal to $k(n)$ and a height h_{max} equal to $2 \times k(n) - 1$ (i.e. the diameter of the coronenoid of size $k(n)$). Figure 7 shows the rectangle benzenoid of dimensions 2×3 embedded in the coronenoid of size 4 (a) and its hexagon graph (b).

Then, starting from model I, we must add new variables to model the desired property. Namely, we add two integer variables x_w and x_h whose domain is respectively $\{1, \ldots, w_{max}\}$ and $\{1, \ldots, h_{max}\}$. These variables represent respectively the number of columns and lines of the built benzenoid. In addition, we denote L_i (resp. C_i) the set of variables x_h which appear in the ith line (resp. ith column) in the coronenoid of size $k(n)$. Likewise, let D_i the set of variables belonging to the ith diagonal of the coronenoid of size $k(n)$. We assume that lines (resp. columns and diagonals) are numbered from top to bottom (resp. from left to right). For example, if we consider the hexagon graph of the coronenoid of size 4, we have the following sets:

$$\begin{cases} L_1 = \{x_1, x_2, x_3, x_4\} \\ L_2 = \{x_5, x_6, x_7, x_8, x_9\} \\ L_3 = \{x_{10}, x_{11}, x_{12}, x_{13}, x_{14}, x_{15}\} \\ L_4 = \{x_{16}, x_{17}, x_{18}, x_{19}, x_{20}, x_{21}, x_{22}\} \\ L_5 = \{x_{23}, x_{24}, x_{25}, x_{26}, x_{27}, x_{28}\} \\ L_6 = \{x_{29}, x_{30}, x_{31}, x_{32}, x_{33}\} \\ L_7 = \{x_{34}, x_{35}, x_{36}, x_{37}\} \end{cases} \quad \begin{cases} C_1 = \{x_1, x_5, x_{10}, x_{16}\} \\ C_2 = \{x_2, x_6, x_{11}, x_{17}, x_{23}\} \\ C_3 = \{x_3, x_7, x_{12}, x_{18}, x_{24}, x_{29}\} \\ C_4 = \{x_4, x_8, x_{13}, x_{19}, x_{25}, x_{30}, x_{34}\} \\ C_5 = \{x_9, x_{14}, x_{20}, x_{26}, x_{31}, x_{35}\} \\ C_6 = \{x_{15}, x_{21}, x_{27}, x_{32}, x_{36}\} \\ C_7 = \{x_{22}, x_{28}, x_{33}, x_{37}\} \end{cases} \quad \begin{cases} D_1 = \{x_{16}, x_{23}, x_{29}, x_{34}\} \\ D_2 = \{x_{10}, x_{17}, x_{24}, x_{30}, x_{35}\} \\ D_3 = \{x_5, x_{11}, x_{18}, x_{25}, x_{31}, x_{36}\} \\ D_4 = \{x_1, x_6, x_{12}, x_{19}, x_{26}, x_{32}, x_{37}\} \\ D_5 = \{x_2, x_7, x_{13}, x_{20}, x_{27}, x_{33}\} \\ D_6 = \{x_3, x_8, x_{14}, x_{21}, x_{28}\} \\ D_7 = \{x_4, x_9, x_{15}, x_{22}\} \end{cases}$$

The desired rectangular shape can be obtained either on the basis of rows and columns or on the basis of columns and diagonals. Note that to switch from one to the other it is sufficient to apply a 60° rotation. So we introduce a new Boolean variable r which is set to true if the rectangular shape is obtained through lines and columns, or false if it is obtained through columns and diagonals. Next, we add several constraints to the general model in order to model the following properties:

- *The hexagons of each line are positioned contiguously* We want to avoid having a Boolean variable equal to 0 between two Boolean variables equal to 1. For the ith line, this can be modeled by imposing a global constraint `regular` over the variables of L_i. For this purpose, we consider the automaton presented in Figure 9.
- *The hexagons of each column are positioned contiguously* We proceed similarly to the lines by considering C_i instead of L_i.
- *The hexagons of each diagonal are positioned contiguously* We proceed similarly to the lines by considering D_i instead of L_i.
- *Lines have a consistent size* Each line must be empty or have a size equal to the current width of the rectangle. The size of a line can be defined as the number of hexagons it contains since we know that all the hexagons are contiguous. For the ith line, we want to add a constraint linking x_w to all the variables in L_i and imposing $\sum_{x_{i_j} \in L_i} x_{i_j} = 0 \vee \sum_{x_{i_j} \in L_i} x_{i_j} = x_w$. However, this constraint makes sense only if lines are exploited (i.e. if r is set to

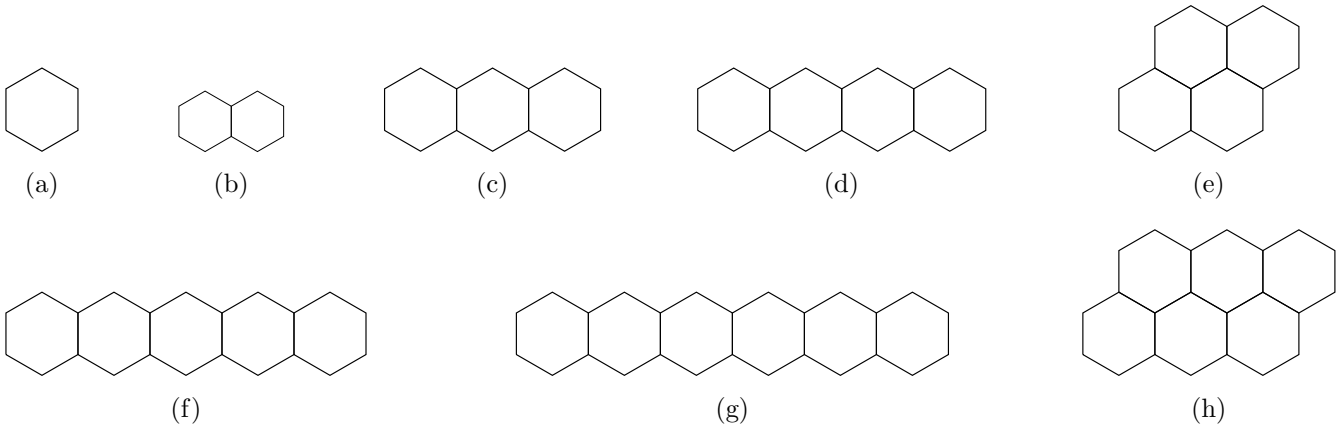


Figure 8: All the rectangular benzenoid structures with $n \leq 6$.

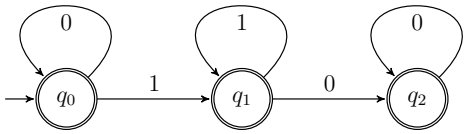


Figure 9: The automaton of the constraints `regular` used in order to impose that lines, columns and diagonals have contiguous hexagons.

true). So we add instead the constraint $r = 1 \Rightarrow \left(\sum_{x_{i_j} \in L_i} x_{i_j} = 0 \vee \sum_{x_{i_j} \in L_i} x_{i_j} = x_w \right)$. As such a constraint is added for each line and we make sure that all the lines have the same width when r is set to true.

- *Columns have a consistent size* We proceed similarly to the lines by considering C_i instead of L_i, except that columns play a different role depending on the value of r. Therefore, we add two constraints per column:

  (i) $r = 1 \Rightarrow \left(\sum_{x_{i_j} \in C_i} x_{i_j} = 0 \vee \sum_{x_{i_j} \in C_i} x_{i_j} = x_h \right)$ and

  (ii) $r = 0 \Rightarrow \left(\sum_{x_{i_j} \in C_i} x_{i_j} = 0 \vee \sum_{x_{i_j} \in C_i} x_{i_j} = x_w \right)$.

- *Diagonal have a consistent size* We proceed similarly to the lines by considering D_i instead of L_i and x_h instead of x_w: $r = 0 \Rightarrow \left(\sum_{x_{i_j} \in D_i} x_{i_j} = 0 \vee \sum_{x_{i_j} \in D_i} x_{i_j} = x_h \right)$.

Now, we focus on benzenoids having a rhombus shape whose chemical properties have been highlighted in [Trinquier and Malrieu, 2018]. A rhombus shape is characterized by its width which can be defined as the number of hexagons in the middle line. As an example, benzene is the benzenoid having a rhombus shape of width 1 while Figure 10 depicts the ones with a width of 2 and 3. We can easily prove that the number of hexagons of a rhombic benzenoid of width w is w^2. Likewise, we can show that a rhombus has w diagonals of hexagons and each diagonal contains exactly w hexagons. So, in this context, a rhombus is a particular kind of rectangle for which the width and the height are both equal to $\lfloor \sqrt{n} \rfloor$. Trivially, if $\sqrt{n}$ is not an integer, the problem has no solution. So modeling the property *"all the built benzenoids have a rhombus shape"*, in addition to the property *"having n hexagons"* can be easily achieved by considering the model for rectangular shape and fixing the value of x_h and x_w to $\lfloor \sqrt{n} \rfloor$ and r to false.

These two extensions of our general model are given as a simple illustration of our approach. Note that we can easily generate benzenoids having a rectangle or rhombus shape with a bespoke algorithm. However, what is interesting in our approach is its flexibility. For instance, if some chemists are interested in identifying the rectangular or rhombic benzenoids which have a given Clar number, we only have to model the property "having a given Clar number" by adding some variables and/or constraints to be able to find the wished benzenoids. The Clar number of a benzenoid is the maximum number of non-adjacent hexagons (i.e. hexagons which have no bond in common) which admit three double bonds [Clar, 1972].

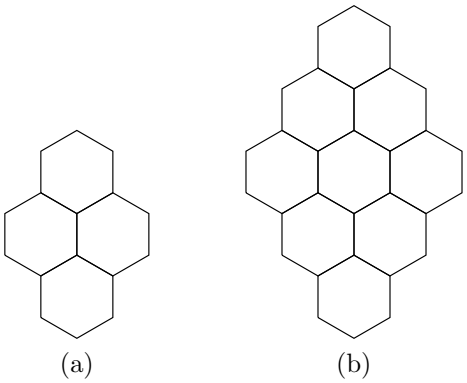


Figure 10: Rhombic benzenoids of width 2 (a) and 3 (b).

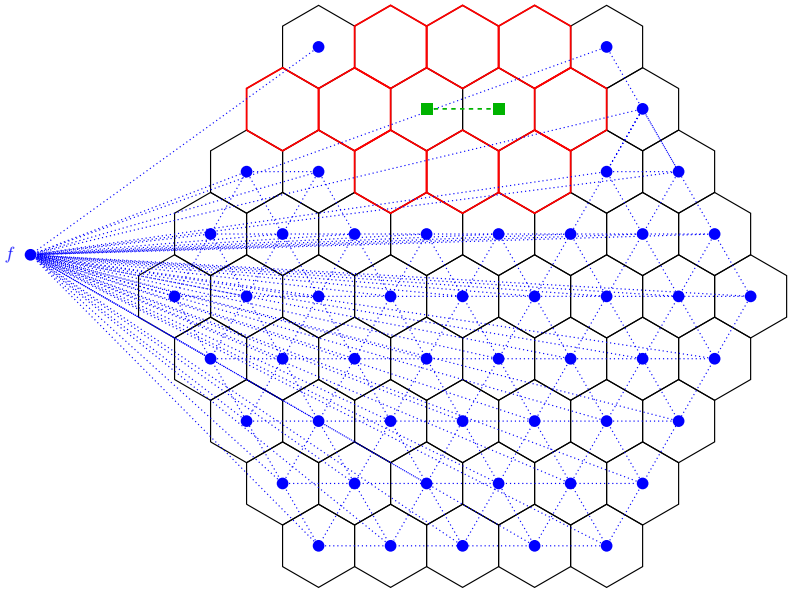


Figure 11: A coronoid of nine hexagons (in red) embedded in the coronenoid of size 5 and its complement (in green and blue). The complement has two connected components (described respectively in blue and green). The first connected component is depicted with blue full circle vertices and dotted edges. It includes the vertex corresponding to the external face (represented by the vertex f) and so does not correspond to a hole. The second component is represented by green square vertices and dashed edges. It does not contain the external face and so represents a hole.

4.4 Generating Coronoids

Chemists refer to benzenoids with at least one hole as *coronoids* (not to be confused with coronenoids). These molecules are promising model structures of graphene with well-defined holes [Di Giovannantonio et al., 2020, Beser et al., 2016, Dias, 2008]. Their enumeration and generation gave rise to several studies (e.g., [Brunvoll et al., 1990] which enumerates 2-hole coronoids and generates the smallest 18 and 19-hexagon 3-hole coronoids). The methods for generating them are quite inefficient or too specific. The first kind of approach tries to build specific kinds of coronoids by considering cycles of hexagons and trying all possible ways to add hexagons. The second kind of approach consists in generating all the benzenoids with n hexagons and then detecting the ones with holes. Another possible approach consists in generating benzenoids without holes (e.g. with the method of Brinkmann et al. [Brinkmann et al., 2002]) and then digging holes in the obtained benzenoids. However, we can note that the two latter approaches can quickly become too time-consuming with respect to an approach that would directly generate coronoids. There are so much fewer coronoids than benzenoids without hole[3] that posting a filtering constraint enforcing holes should be more efficient than a "generate and test" approach. So, in this part, we present how we can model the property *"all the built benzenoids have h holes and are contained in a benzenoid with n hexagons"*. This allows to easily generate all kinds of coronoids with any number of holes.

We start from the idea that a hole is made of hexagons of the embedding coronenoid which do not participate to the

[3]For instance, there exist 6,510 benzenoid structures for 9 hexagons, among which only five are coronoids (see Tables 3 and 5 for more examples).

structure depicted by x_G. Given the hexagon graph $B_h^{c(k(n))}$ of the coronenoid of size $k(n)$, we define $\overline{B_h}$ as the complement of B_h in $B_h^{c(k(n))}$. In other words, the vertices of this graph are the vertices of $B_h^{c(k(n))}$ which do not belong to the current structure B_h (described by x_G) while there still exists an edge between two vertices if the corresponding hexagons share an edge in $B^{c(k(n))}$. We can easily show that any connected component of $\overline{B_h}$ such that no hexagon is located on the edge of $B_h^{c(k(n))}$ corresponds to a hole. Hence we want to use the global constraint `nbConnectedComponents` to constrain the number of holes. Unfortunately, we cannot apply it directly on $\overline{B_h}$ because it can have any number of components that touch an edge of $B_h^{c(k(n))}$ (see Figure 11). However, notice that all the connected components of $\overline{B_h}$ that touch an edge of $B_h^{c(k(n))}$ are in fact connected to the external face of $B_h^{c(k(n))}$. So we just need to add the external face of $B_h^{c(k(n))}$ to form a graph $B_{he}^{c(k(n))}$ thus ensuring that the connected components touching an edge of $B_h^{c(k(n))}$ form a single connected component in $B_{he}^{c(k(n))}$ (i.e. the corresponding hexagons and the external face are part of a single connected component). Therefore, we can constrain the number of connected components of $B_{he}^{c(k(n))}$ to be equal to $h+1$.

In order to fix the number of holes, we add the following variables to the general model I:

- a new graph variable x_H which represents $\overline{B_h}$ and whose vertices are related to the external face and the hexagons that do not belong to x_G. x_H has the same domain as x_G but an additional vertex in its upper bound graph representing the internal face and all the edges linking this vertex to the border hexagons of the coronenoid.
- a set of n_c Boolean variables $\{x_1^H, \ldots, x_{n_c}^H\}$ (with n_c the number of hexagons of the coronenoid of size $k(n)$). Like x_i for x_G, the variable x_i^H is set to 1 if the ith hexagon of the coronenoid of size $k(n)$ is used in the graph depicted by x_H, 0 otherwise. Likewise, we define the set of m_c Boolean variables $y_{i,j}^H$. The variable $y_{i,j}^H$ is set to 1 if the edge $\{i,j\}$ of the coronenoid of size $k(n)$ is used in the graph depicted by x_H, 0 otherwise.

Regarding the constraints, we use channeling constraints again to establish the link between x_i^H, $y_{i,j}^H$ and x_H. Finally, we add the following constraints ensuring that variables x_H and x_C have the right properties:

- x_H *has* $h+1$ *connected components* This is enforced thanks to the graph constraint `nbConnectedComponents` of Choco applied on variable x_H and value $h+1$.
- *A single hexagon does not form a hole* Each vertex/hexagon of x_H must have a degree strictly greater than 0. This constraint eliminates holes that would be a sole hexagon and allows multiple holes. We simply use the `minDegrees` graph constraint of Choco applied on x_H.
- *Any hexagon of the embedding coronenoid is either in* x_G *or in* x_H. We add a xor clause between x_i^G and x_i^H (that is $x_i \oplus x_i^H$) for any $i \in \{1, \ldots, n_c\}$.
- *If a hexagon of* x_H *is in the border of the embedding coronenoid, then there is an edge between this hexagon and the external face.* We add a clause $\overline{x_i^H} \vee y_{i,f}^H$, where f is the index of the external face, for any $i \in \{1, \ldots, n_c\}$ that corresponds to the border.

4.5 Generating Symmetric Benzenoids

Symmetry is a central concept in chemistry at atomic, molecular, and supramolecular levels. In the field of organic chemistry, the quest for highly symmetric compounds with the skeletons of regular polyhedra (tetrahedrons, cubes, and dodecahedrons), pseudo-spherical cages (fullerenes), propellers [Roy et al., 2020], tubes etc. has been the source of intense research [Bauschlicher et al., 2008, Cocchi et al., 2014, Bouwman et al., 2021]. For example, the tetrahedrane molecule, a hypothetical hydrocarbon with a tetrahedral structure has still not been synthesized as of 2021, whereas the synthesis of cubanes and dodecahedranes dates back to the '60s and '80s, respectively [Eaton and Cole, 1964, Ternansky et al., 1982].

Symmetric molecules are not only fascinating due to their intrinsic "beauty", but also because they display a variety of specific properties (electronic, spectroscopic, magnetic, ...) [Kastler et al., 2006, Konishi et al., 2019]. In the field of theoretical chemistry, the fundamental importance of symmetry has long been recognized in the development of modern electronic structure methods. The use of the group theory formalism allows for drastic simplifications in the resolution of the Schrödinger equation, leading to significant savings of computational times [Longuet-Higgins, 1963, Taylor, 1992].

Benzenoids are also classified by chemists by their classes of internal symmetries (symmetries that let a benzenoid invariant by a 60, 120 or 180 degree rotation and/or mirroring). Precisely, chemists have defined 13 types of symmetries taking into account the type of the rotational center: a hexagon, a vertex or an edge (see Figure 12). So a symmetry σ can be defined simply by fixing the rotational center or the mirror axis.

We can generate such classes of benzenoids by adding the constraints for enforcing internal symmetries. More precisely, for each desired symmetry σ, it suffices to post the constraint $x_h \Leftrightarrow x_{\sigma(h)}$ for any hexagon h of x_G, where $\sigma(h)$ is the image of h by the symmetry σ.

However, we must be careful of the location of the symmetry axis (the rotation center or the mirror axis) which is not necessarily at the center of the embedding coronenoid for all symmetrical benzenoids. They can be shifted upward and/or

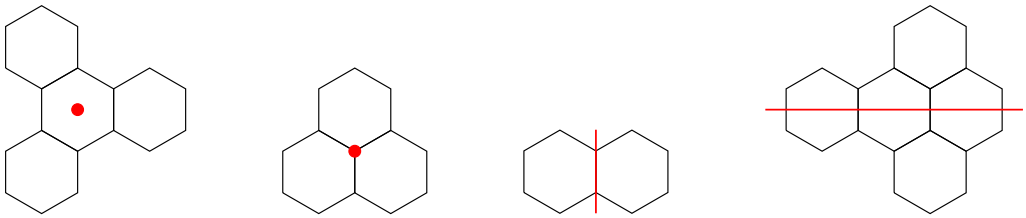


Figure 12: Examples of symmetric benzenoids for which we depict in red the rotational center or the mirror axis: 120° rotation around a hexagon (a) or a vertex (b), and mirror symmetry whose axis is an edge (c) or a hexagon (d).

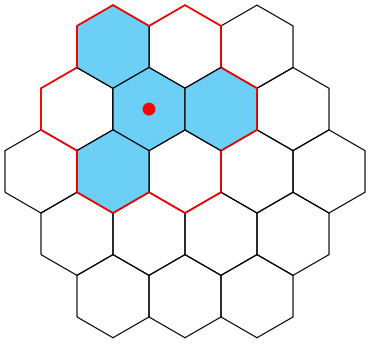


Figure 13: Examples of a not centered symmetric benzenoid (depicted in blue). This benzenoid will be generated in a smaller embedding coronenoid surrounded in red.

to the left if the benzenoid occupies a small area (remember that it is stuck to the top and left borders of the embedding coronenoid). Rather than posting a disjunction of symmetry constraints on the set of all the possible rotational centers or mirror axis, we use the fact that, if the center or the axis is shifted, then the benzenoid could have fit in a smaller embedding coronenoid (see Figure 13). This is why, in the case where we post a symmetry constraint, we fix a single rotation center (or a single mirror axis) but launch the search for symmetrical benzenoids for all the possible sizes lower or equal to $k(n)$ of embedding coronenoids.

4.6 Generating Benzenoid Structures Having a Given Irregularity

In this part, we consider a chemical property called *irregularity parameter* [Bouwman, J. et al., 2019] and describe a CSP model able to generate benzenoid structures having a given value of this parameter. Firstly we have to introduce some definitions :

Definition 1 ([Bouwman, J. et al., 2019]) *Given a benzenoid B and h one of its hexagons, a* group of *the hexagon h is a connected set of carbons of h such that each carbon is linked to one hydrogen. A group involving one carbon (respectively two, three or four carbons) is called a* solo *(resp. a* duo *, a* trio *or a* quarto*). The* size *of a group is the number of carbon atoms it involves.*

Given the fact that hydrogen atoms are not present in our representation of benzenoid structures, we can also define a group by a set of consecutive carbons of a hexagon h having a degree equal to 2. Indeed remember that in a benzenoid, each carbon atom is linked either to two other carbon atoms and one hydrogen atom (and so has a degree of 2) or three other carbon atoms (and so has a degree of 3). For instance, Figure 14 shows examples of a solo (a), a duo (b), a trio (c) and a quarto (d) with the original molecule and its associated graphical representation.

Now, we can define the *parameter of irregularity* of a benzenoid as follows:

Definition 2 ([Bouwman, J. et al., 2019]) *Given a benzenoid B, the* parameter of irregularity *of B, ξ_B is defined as follows :*

$$\xi_B = \frac{N_3+N_4}{N_1+N_2+N_3+N_4}$$

with N_i representing the number of carbons that belong to a group of size i.

If we consider phenanthrene depicted on Figure 15, we can easily see that it has 4 quartos and one duo. So, we have $N_2 = 2$, $N_4 = 8$ (because there are two carbons that belong to a duo and eight to a quarto), and $N_1 = N_3 = 0$ (because the phenanthrene does not contain any solo or trio). Therefore, we have $\xi = \frac{8}{10} = 0.8$.

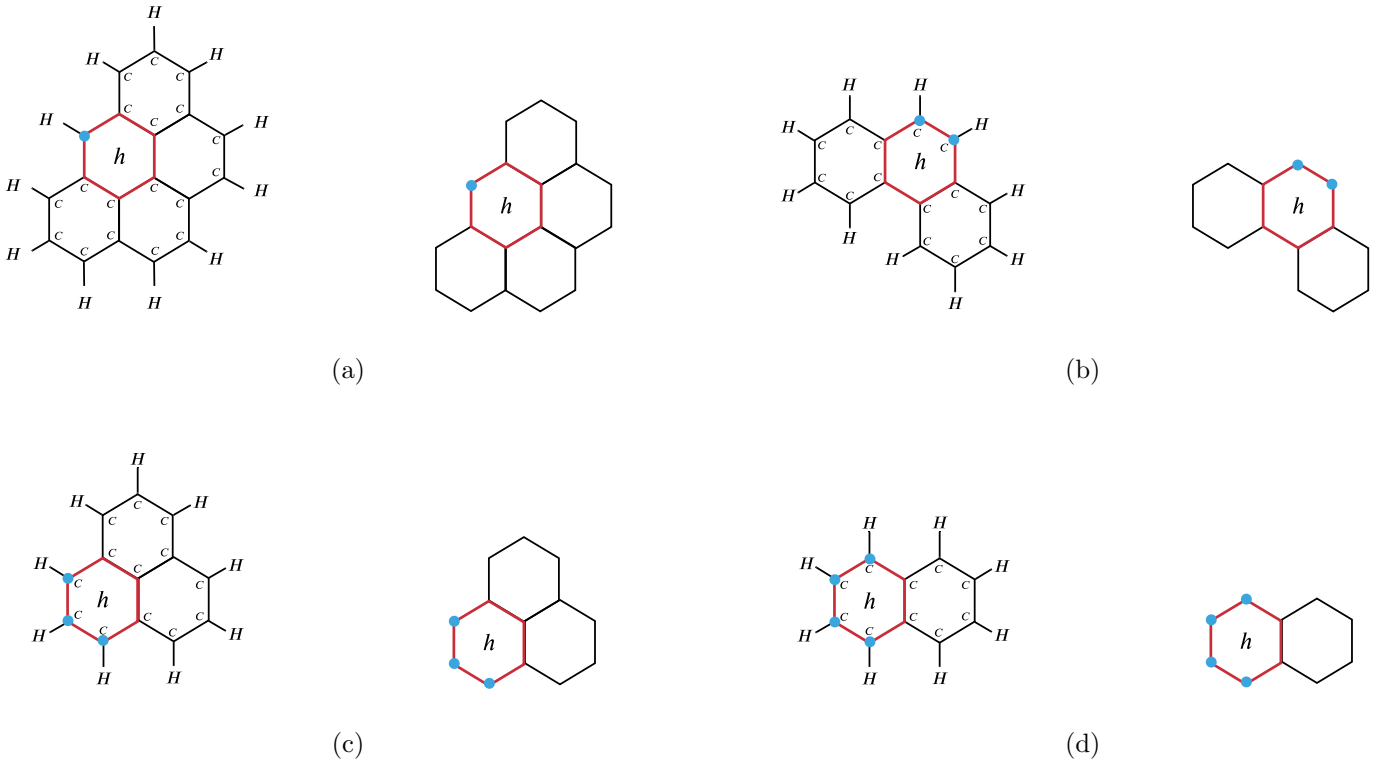


Figure 14: Examples of a solo (a), a duo (b), a trio (c) and a quarto (d).

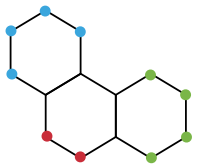
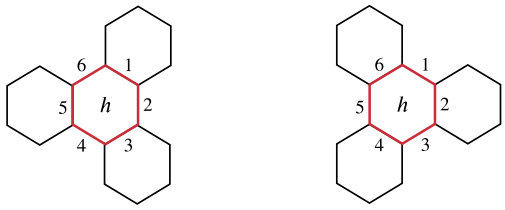


Figure 15: Groups of phenanthrene.

Chemists exploit this parameter to characterize the irregularity of PAHs. For instance, in [Bouwman, J. et al., 2019], Bouwman et al. presented the results of the computation of infrared spectrums of PAHs classified according to their parameter of irregularity. Generating benzenoid structures having a given parameter of irregularity (or included in an interval) could be interesting from a chemistry point of view. For example, it could be useful in order to complete the work of Bouwman et al. by focusing on benzenoid structures having a given irregularity parameter.

Before describing how this property can be modeled as a CSP instance, we highlight two of its features. Firstly, we can note that, given a hexagon h, the existence of solo, duo, trio or quarto in h only depends on the presence of hexagons around h. In other words, the hexagons which appear in the neighborhood of h in the hexagon graph allow us to determine if a solo, a duo, a trio or a quarto occurs or not in h. In order to formally exploit this feature, we first label the positions of the six possible neighbors of h as follows. Position 1 is associated with the top-right neighbor while the other positions are successively labeled from 2 to 6 in a clockwise manner. Figure 16 provides an illustration of this labeling. Then, we call *neighborhood configuration of* h a 6-tuple such that the ith element is equal to 1 if h has a neighbor at the ith position, 0 otherwise. For instance, Figure 16(a) shows us a hexagon h having $(1, 0, 1, 0, 1, 0)$ as neighborhood configuration. With this configuration, h clearly does not admit any solo, duo, trio or quarto. Secondly, we can observe that if we switch from one neighborhood configuration to another by a 60 degree rotation, these two configurations have the same numbers of solo, duo, trio and quarto. So, given a neighborhood configuration N_h of a hexagon h, we denote N_h^* the set of all the neighborhood configurations that can be obtained by applying rotations on N_h. For instance, Figure 16 shows the set of configurations $(1, 0, 1, 0, 1, 0)^*$. We can also remark that applying a 60 degree rotation on a configuration is equivalent to making a cyclic permutation on the associated tuple. So we can easily identify all the possible N_h^* and for each, determine the existing groups. This work must be achieved once. Figure 17 presents the 13 possible configurations N_h^* and the



(a) (b)

Figure 16: The set of neighborhood configurations $(1,0,1,0,1,0)^*$.

corresponding groups. Note that case (a) is a particular case where h has a group having a size greater than 4. As we are only interested in groups of sizes 1 to 4, we consider that it induces no group. Nevertheless, this case has little interest since it only occurs for a single benzenoid, namely benzene.

As for the previous models, we start from the general model and add some variables and constraints in order to be able to generate benzenoids having a given irregularity parameter. Firstly, we add the following variables (or sets of variables) to the general model:

- $X_Z = \{z_1, \ldots, z_{n_c}\}$: a set of Boolean variables ($\{0,1\}$ as domain). The variable z_h is equal to 1 if the hexagon h induces no group, 0 otherwise.
- $X_S = \{s_1, \ldots, s_{n_c}\}$: a set of integer variables with the domain $\{0,1,2\}$. The variable s_h is equal to the number of carbons of the hexagon h that belong to a solo.
- $X_D = \{d_1, \ldots, d_{n_c}\}$: a set of integer variables with the domain $\{0,2\}$. The variable d_h is equal to the number of carbons of the hexagon h that belong to a duo.
- $X_T = \{t_1, \ldots, t_{n_c}\}$: a set of integer variables with the domain $\{0,3\}$. The variable t_h is equal to the number of carbons of the hexagon h that belong to a trio.
- $X_Q = \{q_1, \ldots, q_{n_c}\}$: a set of integer variables with the domain $\{0,4\}$. The variable q_h is equal to the number of carbons of the hexagon h that belong to a quarto.
- N_0: an integer variable of domain $\{0, \ldots, n_c\}$ that represents the number of hexagons that do not induce any group.
- N_1, N_2, N_3, N_4: four integer variables that respectively represent the number of solos, duos, trios and quartos over all the hexagons. Each variable has $\{0, \ldots, n_c\}$ as domain.
- x_ξ: an integer variable of domain $\{0, 1, \ldots, 100\}$ that represents the parameter of irregularity multiplied by 100. We have to do so because Choco does not handle variables with real numbers. By so doing we have:

$$x_\xi = 100 \times \xi = \frac{100 \times (N_3 + N_4)}{N_1 + N_2 + N_3 + N_4}$$

  By considering the value 100, we assume that an accuracy of 0.01 is sufficient. If this is not the case, it is naturally possible to adapt this value according to the desired precision.

Regarding the constraints, the first task consists in expressing the relationship between the existence of h (namely the variable x_h) and the associated variables z_h, s_h, d_h, t_h and q_h. If the hexagon h does not exist (i.e. x_h is equal to 0), by definition, there cannot exist a group of any size. To model this, we consider the following constraints:

$$\begin{cases} x_h = 0 \Rightarrow z_h = 0 \\ x_h = 0 \Rightarrow s_h = 0 \\ x_h = 0 \Rightarrow d_h = 0 \\ x_h = 0 \Rightarrow t_h = 0 \\ x_h = 0 \Rightarrow q_h = 0 \end{cases}$$

In contrast, if the hexagon h exists (and so $x_h = 1$), we have to compute the values of z_h, s_h, d_h, t_h and q_h depending on the neighborhood configuration of h. In order to do so, we consider a table constraint[4] whose allowed tuples are built from

[4] *Table constraints* list explicitly the allowed (or disallowed) combinations of values that a specific set of variables can take [Lecoutre, 2009].

Neighborhood configuration of h						z_h	s_h	d_h	t_h	q_h
1	0	0	0	0	0	0	0	0	0	1
0	1	0	0	0	0	0	0	0	0	1
0	0	1	0	0	0	0	0	0	0	1
0	0	0	1	0	0	0	0	0	0	1
0	0	0	0	1	0	0	0	0	0	1
0	0	0	0	0	1	0	0	0	0	1

Table 1: Part of the table constraint associated to the set of configuration $(1,0,0,0,0,0)^*$ represented on Figure 17(b) for the hexagon h.

the possible cases listed in Figure 17. In other words, we have an allowed tuple per possible neighborhood configuration and each allowed tuple maps a configuration to the corresponding number of groups for each group. For instance, Table 1 presents the part of the table constraint which is associated to the set of configuration $(1,0,0,0,0,0)^*$ represented on Figure 17(b) for a hexagon h. Regarding the scope of the constraint, it involves all the variables x_i corresponding to hexagons in the neighborhood of h and the variables z_h, s_h, d_h, t_h and q_h. This table constraint is only enabled when x_h is set to 1. In Choco solver, it can be easily achieved thanks to the `ifThen` constraint. Note that hexagons located on a border of the hexagon graph of the coronenoid of size $k(n)$ cannot have their six neighbors. In such a case, we only consider the existing neighbors in the scope and the table is computed by selecting the rows having the value 0 for the missing hexagons and by projecting the obtained relation over the constraint scope.

With these constraints, we are able to know all the groups for each hexagon according to its neighborhood configuration. We have to add constraints in order to set the values of N_0, N_1, N_2, N_3 and N_4 by using several sum constraints:

- N_0 *is equal to the number of hexagon that do not induce any groups*: $N_0 = \sum_{z_h \in X_Z} z_h$
- N_1 *is equal to the number of solos*: $N_1 = \sum_{s_h \in X_S} s_h$
- N_2 *is equal to the number of duos*: $N_2 = \sum_{d_h \in X_D} d_h$
- N_3 *is equal to the number of trios*: $N_3 = \sum_{t_h \in X_T} t_h$
- N_4 *is equal to the number of quartos*: $N_4 = \sum_{q_h \in Q} q_h$

Then, we have to set the value of x_ξ to $\frac{100\times(N_3+N_4)}{N_1+N_2+N_3+N_4}$. It can be easily done by applying some predicate constraints, that is $x_\xi = 100 \times (N_3 + N_4) \div (N_1 + N_2 + N_3 + N_4)$ where $\div$ represents the quotient of the Euclidean division.

To conclude, the model can add several arithmetic constraints $(\leq, =, \geq)$ on $x_\xi, N_0, N_1, N_2, N_3$ and N_4 according to the user's wishes. For example, it is able to generate benzenoid structures having a given parameter of irregularity and/or a given number of solos/duos/trios/quartos.

4.7 Generating Benzenoid Structures Having Given Numbers of Carbon and Hydrogen Atoms

In this part, we describe a CSP model able to generate benzenoid structures having a given number of carbon and hydrogen atoms. Being able to generate molecules having this property is interesting from a chemical viewpoint. Indeed, chemists usually refer to molecules by giving their *molecular formula*. The molecular formula specifies the number of atoms of each element appearing in the molecule. For instance, anthracene described in Figure 1 has $C_{14}H_{10}$ as molecular formula meaning that it consists of 14 carbon atoms and 10 hydrogen atoms. For an illustration, in the context of PAHs, chemists exploit the molecular formula to classify PAHs (e.g. in the NASA Ames PAH database [Bauschlicher et al., 2018]). Moreover, they often deal with *isomers* that are molecules having the same molecular formula but different structures.

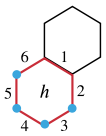
Given two integers ν and μ representing the wanted numbers of carbon and hydrogen atoms, we start from the general model and add some variables and constraints in order to be able to generate benzenoid structures having a given number of carbon and hydrogen atoms. We also describe next how to constrain the numbers of hydrogen and carbon atoms.

4.7.1 Constraining the Number of Hydrogen Atoms

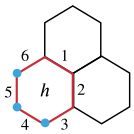
In order to constrain the number of hydrogen atoms, we first need to express this number in our model. In order to do so, we can note that the number of hydrogen atoms is equal to the number of carbon atoms involved in a solo, a duo, a trio



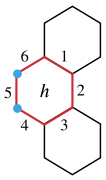
(a) $(0,0,0,0,0,0)^* \implies$ no group



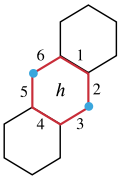
(b) $(1,0,0,0,0,0)^* \implies$ a quarto



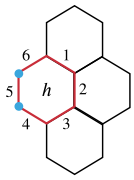
(c) $(1,1,0,0,0,0)^* \implies$ a trio



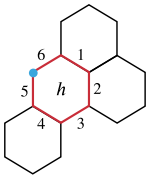
(d) $(1,0,1,0,0,0)^* \implies$ a duo



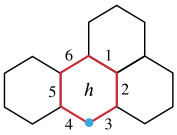
(e) $(1,0,0,1,0,0)^* \implies$ two solos



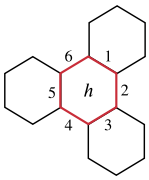
(f) $(1,1,1,0,0,0)^* \implies$ a duo



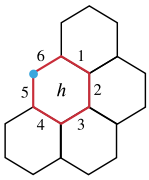
(g) $(1,1,0,1,0,0)^* \implies$ a solo



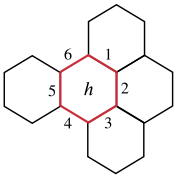
(h) $(1,1,0,0,1,0)^* \implies$ a solo



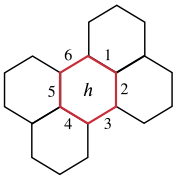
(i) $(1,0,1,0,1,0)^* \implies$ no group



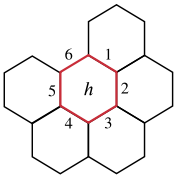
(j) $(1,1,1,1,0,0)^* \implies$ a solo



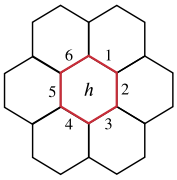
(k) $(1,1,1,0,1,0)^* \implies$ no group



(l) $(1,1,0,1,1,0)^* \implies$ no group

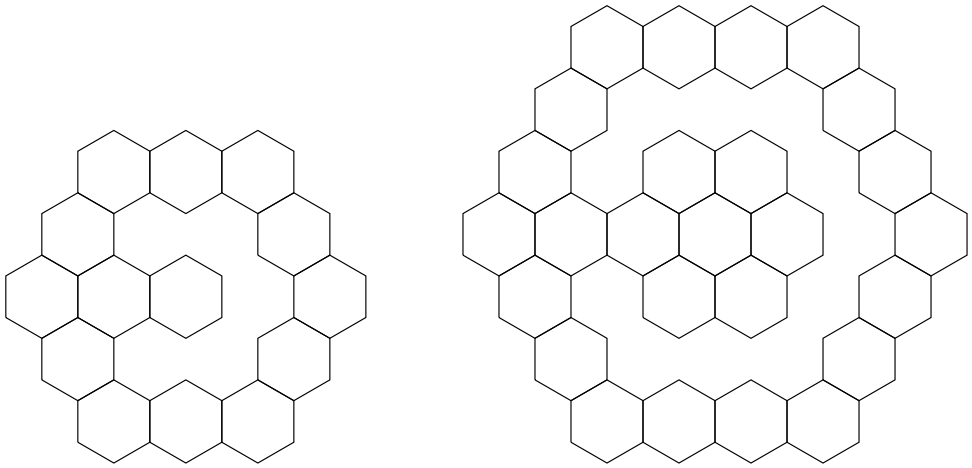


(m) $(1,1,1,1,1,0)^* \implies$ no group



(n) $(1,1,1,1,1,1)^* \implies$ no group

Figure 17: All the induced groups for all the neighborhood configurations.



(a) (b)

Figure 18: Benzenoid structures maximizing the number of hydrogens and which can be embedded in a coronenoid of size 3 (a) and 4 (b).

or a quarto as soon as we consider benzenoids having at least two hexagons. In the particular case of a single hexagon (corresponding to benzene), the number of hydrogen atoms is six. So, we introduce an integer variable N_H in order to represent the number of hydrogen atoms. We now define its domain with tight bounds. Considering that the smallest number of hydrogen atoms in a benzenoid is reached for benzene, this domain can start from 6. Regarding the largest number, it is obtained for benzenoid structures built as follows. We start from benzene if $k(n)$ is odd, from the coronenoid of size two otherwise. Then, we alternatively add an empty crown and a crown of hexagons until we have $k(n)$ crowns. We also add a hexagon in each empty crown to ensure the connectedness of benzenoids. Figure 18 presents the obtained benzenoids when $k(n)$ is equal to 3 or 4. Then, we can easily see that a crown of size c has $6 \times c$ hydrogens in its outer border and $(c-1) \times c$ in its inner border. Therefore, by considering the hexagons added to ensure the connectedness, we can define the upper bound of the domain of N_H thanks to the following formula:

$$\begin{cases} 6 & \text{if } k(n) = 1 \\ 4 + \sum\limits_{1<c\leq k(n),\ c \text{ is odd}} 6 \times c + (c-1) \times c - 2 & \text{if } k(n) \text{ is odd and greater than 1} \\ 10 + \sum\limits_{2<c\leq k(n),\ c \text{ is even}} 6 \times c + (c-1) \times c - 2 & \text{otherwise} \end{cases}$$

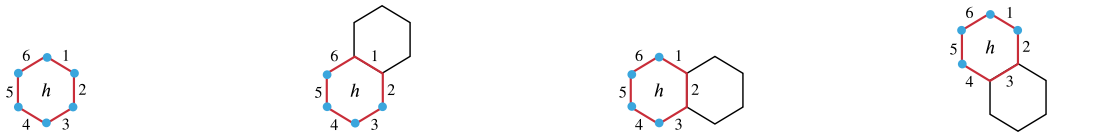
If n is equal to 1, we directly fix N_h to 6. Otherwise, the value of N_H relies on the number of carbon atoms involved in a solo, a duo, a trio or a quarto. In order to do so, we add all the variables defined in the subsection 4.6 except the variable x_ξ. Of course, we also consider all the related constraints. Then, we establish the link between N_H and N_1, N_2, N_3 and N_4 by posting the constraint $N_H = N_1 + N_2 + N_3 + N_4$.

Finally, we constrain the number of hydrogen atoms by adding the adequate constraint. Indeed, similarly to the irregularity, various arithmetic constraints can be used depending on the needs of chemists. For instance, if we want to generate benzenoid structures having μ hydrogen atoms, we add the constraint $N_H = \mu$.

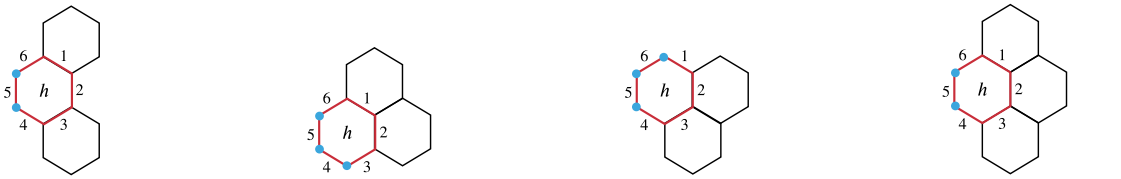
4.7.2 Constraining the Number of Carbon Atoms

Our general model mainly relies on hexagons and so carbon atoms (like hydrogen atoms) are not explicitly represented. Moreover, some carbon atoms can be shared by two or three hexagons. Hence, when modeling the number of carbon atoms, we must be careful not to count the same carbon atom several times. In order to avoid this problem, we exploit a partition of the set of carbon atoms based on the hexagons. A carbon atom is attached to a hexagon h if it only appears in h, or it appears in at least two hexagons (including h) and h is the leftmost. This can be easily implemented by considering, for each hexagon, the part of its neighborhood located on the right and using neighborhood configuration in a similar way to what was done for the irregularity in subsection 4.6. In other words, the neighborhood is restricted to the hexagons 1, 2 and 3 (that is top-right, right and bottom-right hexagons). Figure 19 shows all the possible configurations for a hexagon h and the associated number of carbon atoms according to the defined partition. For instance, Figure 20 describes the number of carbon atoms for each hexagon of coronene according to the defined partition.

This can be modeled, from the general model, by introducing first an integer variable N_C in order to represent the number of carbons of the benzenoid structure. Its domain is $\{6, \ldots, \nu_{k(n)}\}$ with $\nu_{k(n)}$ the number of carbon atoms of the coronenoid of size $k(n)$. We can easily prove by induction on $k(n)$ that $\nu_{k(n)}$ is equal to $6.k(n)^2$. Then, we also add to the



(a) 6 carbon atoms (b) 4 carbon atoms (c) 4 carbon atoms (d) 4 carbon atoms



(e) 2 carbon atoms (f) 3 carbon atoms (g) 3 carbon atoms (h) 2 carbon atoms

Figure 19: All the possible neighborhood configuration for a hexagon h and the associated number of carbon atoms according to the defined partition.

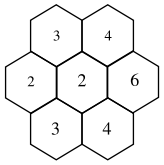


Figure 20: The number of carbon atoms for each hexagon of coronene according to the defined partition.

general model several integer variables $N_C^1, \ldots, N_C^{n_c}$ with domain $\{2, 3, 4, 6\}$. The variable N_C^h corresponds to the number of carbon atoms attached to the hexagon h according to the defined partition. Regarding constraints, we consider the table constraint described in Table 2. Each tuple of this table represents one of the configurations described in Figure 19. As for the scope of the constraint and a hexagon h, it involves all the variables x_i corresponding to the neighbor hexagons at the right of h and N_C^h. Like previously, this table constraint is only enabled when $x_h = 1$ (done again with an *ifThen* constraint). Clearly, if h is located on the top, right or bottom border of the coronenoid of size $k(n)$, we adapt the constraint like in the previous subsection. Finally, we define N_C by summing the number of carbon atoms of each hexagon thanks to the constraint $N_C = \sum_{i \in \{1, \ldots n_c\}} N_c^i$.

Once the number of carbon atoms is expressed, we can constrain it by adding the adequate constraint depending on the needs of chemists. For instance, if we want to generate benzenoid structures having ν carbons atoms, we can add the constraint $N_C = \nu$.

Configuration of h			c_h
0	0	0	6
0	0	1	4
0	1	0	4
0	1	1	3
1	0	0	4
1	0	1	2
1	1	0	3
1	1	1	2

Table 2: Table constraint corresponding to the possible neighborhood configurations and their associated carbons contributions.

4.8 Combining Some Models

In this part, we discuss the possibility of combining several properties. Combining several properties is clearly a need expressed by chemists. Notably, they often consider symmetrical molecules with some additional properties (e.g. [Bauschlicher et al., 2008, Mishra et al., 2018, Sánchez-Grande et al., 2021, Silva and Girão, 2021, Konishi et al., 2019]). One of the main interests of our approach based on constraint programming is its flexibility. As mentioned previously, we can easily model new properties by adding some variables and/or constraints. It is the same for combining different properties to generate desired benzenoid structures. Generally, combining two properties only consists in merging the two corresponding models. This is made possible by the fact that all our models are based on the same general model. For instance, by doing so, we can combine the symmetric benzenoid property with any other considered structural property. In some cases, such combinations are of little interest. For example, we can combine the model of catacondensed benzenoids with one of rectangular benzenoids but we already know that there exists a single solution (i.e. the rectangle $1 \times n$). Likewise, we know in advance that combining the model of rectangular benzenoids with one of coronoids leads to a problem with no solution since the rectangle must be full.

5 Generating Benzenoid Structures in Practice

In this section, we assess the ability of our model to produce benzenoid structures depending on the desired properties from a practical viewpoint. With this aim in view, we implement our model in Choco Solver (v. 4.10.7) and use its default setting. This generator of benzenoid structures is a part of our BenzAI software[5]. The experiments are carried out on Dell PowerEdge M610 with processors Intel Xeon E5620 2.4 GHz and 24 GB of memory. A single thread is run for each generation with at most 12 GB of memory.

First, we consider our general model. We vary the number n of hexagons from 1 to 10 by steps of 1. Table 3 provides the number of variables and constraints of the model, the number of produced benzenoid structures (that is the number of found solutions) and the runtime in seconds. The first observation is that the numbers of variables and constraints are the same for any consecutive integer n and $n+1$ when n is even. This is explained by the fact that these two numbers are directly related to the size $k(n)$ of the considered coronenoid and, by definition, $k(n) = k(n+1)$ if n is even. Moreover, Choco solver stores all the clauses (e.g. those for the lex-lead constraint or for the nogoods) in a single constraint which is handled as a clause base. Then, we can also note that solving the problem for $n+1$ hexagons is more time-expensive than solving the problem for n hexagons even if these two problems are really close in terms of variables and constraints. Indeed, the main differences consist of a larger clause base and a more relaxed constraint about the number of hexagons, which requires traversing a larger part of the search space. Finally, we can remark that the runtime for generating all the benzenoid structures with 10 hexagons starts to become important since this runtime exceeds 8 hours. Nonetheless, this is not necessarily a hindrance because generating all structures only makes sense for chemists up to 9 hexagons. Beyond that, the number of structures becomes too large (it increases by about 4.5 for each added hexagon) to be usable in practice. In practice, chemists are more interested in generating structures satisfying some properties. As seen before, the main advantage of our approach is its flexibility. The general model can be easily specialized by adding some variables and constraints, which generally allows us to reduce the size of the search space to explore and so to obtain a reasonable runtime.

Table 4 presents the results obtained when generating all catacondensed benzenoid structures. As we can see, the simple addition of the `tree` constraint leads to a significant decrease in computation time. It is the same for the generation of coronoids (see Table 5) even if this specialization requires the addition of a new graph variable and its associated Boolean variables for expressing its vertices. Finally, chemists are often looking for symmetrical structures. This feature makes it possible to reduce significantly the search space to explore and so the runtime (see Tables 6-8 for instances). Moreover, as mentioned previously, some symmetries allow us to consider a smaller coronenoid and thus to significantly reduce the size of considered instances and their solving runtime. For example, this is the case for 60° rotation as we can see in Table 6. As an illustration, we can note that, for 17 hexagons, we have to only consider a coronenoid of size 4 instead of a coronenoid of size 9 usually.

[5]BenzAI is an open source software for chemists that includes the work presented in this article (generation of benzenoid structures and estimation of the aromaticity) in a user-friendly graphical interface. More information and source code can be found at https://benzai-team.github.io/BenzAI/.

n	$k(n)$	$\|X\|$	$\|C\|$	#structures	runtime
3	2	113	30	3	0.11
4	3	287	90	7	0.21
5	3	287	90	22	0.35
6	4	551	186	81	1.13
7	4	551	186	331	3.29
8	5	905	318	1,436	38.62
9	5	905	318	6,510	466.31
10	6	1,349	486	30,129	29,958.01

Table 3: The size $k(n)$ of the embedding coronenoid, the number of variables and constraints of the general model, the number of produced benzenoid structures and the runtime (in seconds) for the general model when the number n of hexagons varies from 3 to 10.

n	$k(n)$	$\|X\|$	$\|C\|$	#structures	runtime
3	2	113	31	2	0.11
4	3	287	91	5	0.20
5	3	287	91	12	0.28
6	4	551	187	36	0.73
7	4	551	187	118	1.72
8	5	905	319	411	12.83
9	5	905	319	1,489	47.88
10	6	1,349	487	5,572	1,342.69

Table 4: The size $k(n)$ of the embedding coronenoid, the number of variables and constraints of the general model, the number of produced benzenoid structures and the runtime (in seconds) for the catacondensed model when the number n of hexagons varies from 3 to 10.

6 Computing the Resonance Energy of a Benzenoid

6.1 Definitions

6.2 Computing the Resonance Energy

6.2.1 Computing Resonance Energy by Enumerating Kekulé Structures

In this part, we represent a hexagon by a 6-tuple of vertices $h = (v_1, v_2, v_3, v_4, v_5, v_6)$ in which the vertices are listed in a clockwise manner from the vertex at the top of h. In other words, v_1 refers to the vertex at the top of the hexagon, v_2 to the top-right vertex, etc.

In 1999, Lin and Fan [Lin and Fan, 1999] proposed a method able to estimate the aromaticity of benzenoids based on the definition of the resonance energy. Given a benzenoid B, this method (described in Algorithm 1) first enumerates all minimal conjugated circuits (i.e. computes a h-MCC for all its hexagons h) for each Kekulé structure. Then, it deduces the energy induced by each minimal conjugated circuit and adds them up. Finally, it divides the obtained sum by the number of Kekulé structures to obtain the resonance energy. Such a method was implied in Randić's work. The main contribution of Lin and Fan consists in describing how to compute the h-MCCs.

To this end, given a benzenoid B, a Kekulé structure K and h a hexagon of B, Lin and Fan carefully identify all the specific forms of the h-MCC depending on the location of the double bonds of h. For each configuration of double bonds

n	#crowns	$\|X\|$	$\|C\|$	#structures	runtime
8	3	376	113	1	0.91
9	3	376	113	5	1.08
10	4	730	227	43	47.89
11	4	730	227	283	159.93
12	5	1,206	383	1,954	9,900.50
13	5	1,206	383	12,364	182,386.19

Table 5: The size #crowns of the embedding coronenoid, the number of variables and constraints of the general model, the number of produced benzenoid structures and the runtime (in seconds) for the coronoid model when the number n of hexagons varies from 8 to 13.

n	#crowns	$\sum \lvert X \rvert$	$\max \lvert X \rvert$	$\sum \lvert C \rvert$	$\max \lvert C \rvert$	#structures	runtime
3	2	111	111	29	29	0	0.05
4	2	111	111	29	29	0	0.05
5	2	111	111	29	29	0	0.05
6	2	111	111	29	29	0	0.06
7	2	113	113	30	30	1	0.10
8	3	396	285	118	89	0	0.08
9	3	396	285	118	89	0	0.08
10	3	396	285	118	89	0	0.09
11	3	396	285	118	89	0	0.09
12	3	398	287	119	90	1	0.15
13	3	398	287	119	90	2	0.16
14	4	945	549	303	185	0	0.16
15	4	945	549	303	185	0	0.12
16	4	945	549	303	185	0	0.18
17	4	945	549	303	185	0	0.15

Table 6: The size #crowns of the largest considered embedding coronenoid, the total and maximal number of variables and constraints of the general model, the number of produced benzenoid structures and the runtime (in seconds) for the symmetric benzenoid model (60° rotation) when the number n of hexagons varies from 3 to 17.

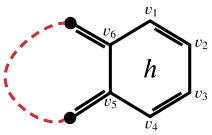
n	#crowns	$\sum \lvert X \rvert$	$\max \lvert X \rvert$	$\sum \lvert C \rvert$	$\max \lvert C \rvert$	#structures	runtime
3	2	111	111	29	29	0	0.05
4	2	113	113	30	30	1	0.10
5	3	396	285	118	89	0	0.08
6	3	396	285	118	89	0	0.09
7	3	400	287	120	90	3	0.16
8	4	945	549	303	185	0	0.15
9	4	945	549	303	185	0	0.17
10	4	949	551	305	186	9	0.27
11	5	1,848	903	620	317	0	0.40
12	5	1,854	905	623	318	2	0.56
13	5	1,854	905	623	318	32	1.19
14	6	3,195	1,347	1,105	485	0	1.44
15	6	3,203	1,349	1,109	486	7	2.20
16	6	3,203	1,349	1,109	486	130	5.18
17	7	5,076	1,881	1,794	689	0	10.94

Table 7: The size #crowns of the largest considered embedding coronenoid, the total and maximal number of variables and constraints of the general model, the number of produced benzenoid structures and the runtime (in seconds) for the symmetric benzenoid model (120° rotation) when the number n of hexagons varies from 3 to 17.

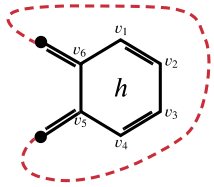
of h, they compute paths of minimal size whose edges correspond alternately to single and double bonds between some vertices of h (denoted *minimal conjugated path* in this part). Note that a minimal conjugated path between two vertices of h must not contains any edge of h. Figure 21 lists all the double bond configurations and their associated h-MCC forms identified by Lin and Fan. For instance, if $h = (v_1, v_2, v_3, v_4, v_5, v_6)$ has the double bond configuration represented in the configuration 2 of Figure 21, Lin and Fan build the minimal circuit of h by finding a minimal conjugated path between v_4 and v_1 and adding it to the path described by $v_1 - v_2 - v_3 - v_4$ (or $v_1 - v_6 - v_5 - v_4$), where $v_1 - v_2 - v_3 - v_4$ denotes the path using the edges $\{v_1, v_2\}$, $\{v_2, v_3\}$ and $\{v_3, v_4\}$. Note that the concatenation of two paths is denoted by the operator $\oplus$ in Algorithm 1.

Algorithm 1 describes this method. Firstly, given a hexagon h and two of its vertices u and v, the procedure *min_path* computes a minimal conjugated path between u and v. Secondly, given a Kekulé structure K and a hexagon h, the function $double_bond_configuration$ returns the double bond configuration of h in K, that is an integer between 1 and 4 according to the four configurations described in Figure 21. The function *min_size* returns the circuit having the smallest size among the circuits given as input.

The main drawback of this method is that it requires generating all the Kekulé structures of the benzenoid. As the number of Kekulé structures may be exponential, this method is clearly inefficient in practice and can only be used for benzenoids with a small number of Kekuké structures.

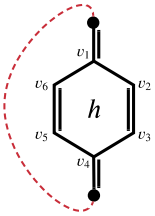


(a)

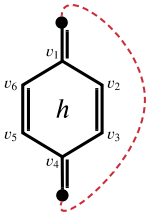


(b)

Configuration 1

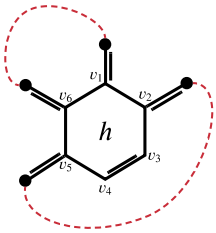


(c)

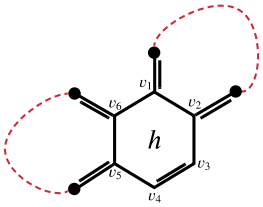


(d)

Configuration 2

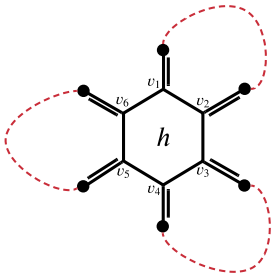


(e)

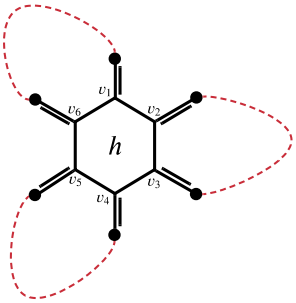


(f)

Configuration 3



(g)



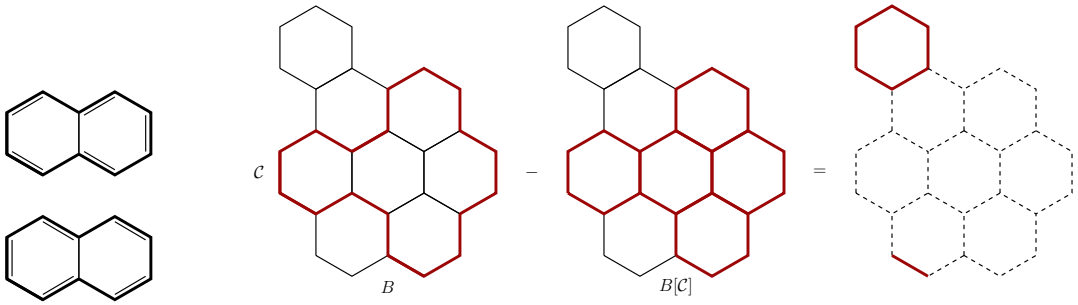
(h)

Configuration 4

Figure 21: All the double bonds configurations for a hexagon [Lin and Fan, 1999].

n	#crowns	$\sum \lvert X \rvert$	$\max \lvert X \rvert$	$\sum \lvert C \rvert$	$\max \lvert C \rvert$	#structures	runtime
3	2	113	113	30	30	1	0.10
4	3	396	285	118	89	0	0.08
5	3	400	287	120	90	2	0.13
6	4	945	549	303	185	0	0.15
7	4	951	551	306	186	6	0.28
8	5	1,848	903	620	317	0	0.43
9	5	1,854	905	623	318	20	0.81
10	6	3,203	1,349	1,109	486	1	1.87
11	6	3,203	1,349	1,109	486	72	4.38
12	7	5,086	1,883	1,799	690	5	13.53
13	7	5,086	1,883	1,799	690	304	42.56
14	8	7,593	2,507	2,729	930	16	143.36
15	8	7,593	2,507	2,729	930	1341	626.74
16	9	10,814	3,221	3,935	1,206	108	686.64
17	9	10,814	3,221	3,935	1,206	342	345.60

Table 8: The size $k(n)$ of the embedding considered coronenoid, the number of variables and constraints of the general model, the number of produced benzenoid structures and the runtime (in seconds) for the symmetric benzenoid model (180° rotation) when the number n of hexagons varies from 3 to 17.



(a) (b)

Figure 22: A cycle $\mathcal{C}$ such as $M(\mathcal{C}) = 2$ (a) and an example of computation of the number of occurrences of a cycle (b).

6.2.2 Computing Resonance Energy Without Enumerating Kekulé Structures

To overcome the issues described in the previous part, Lin [Lin, 2000] proposed another method that is able to compute all the minimal circuits of a given benzenoid without generating its Kekulé structures. This method only considers circuits having a size at most 4. So, it provides an approximation of the resonance energy. Before describing this method, let us introduce some needed definitions.

Definition 3 *Let B be a benzenoid and $\mathcal{C}$ a cycle of B (with $4i+2$ edges, $i \in \mathbb{N}$). $M(\mathcal{C})$ is the number of perfect matchings of $\mathcal{C}$ and its interior inducing a minimal circuit for at least one of the hexagons it covers.*

For example, let us consider $\mathcal{C}$ as being the cycle of size 2 represented in Figure 22(a). $\mathcal{C}$ can induce two different conjugated circuits that are clearly minimal and cover the two hexagons. So we have $M(\mathcal{C}) = 2$.

Definition 4 ([Randić et al., 1996]) *Let $B = (V, E)$ be a benzenoid and $\mathcal{C}$ a cycle of B (with $4i + 2$ edges, $i \in \mathbb{N}$). $B[\mathcal{C}]$ is the sub-graph of B induced by $\mathcal{C}$ and its interior.*

The method presented by Lin [Lin, 2000] relies on the following theorem:

Theorem 1 ([Randić et al., 1996]) *Let B be a benzenoid and $\mathcal{C}$ a cycle of B (with $4i+2$ edges, $i \in \mathbb{N}$). $\mathcal{C}$ is a h-MCC in $\lvert \mathcal{K}(B - B[\mathcal{C}]) \rvert \times M(\mathcal{C})$ Kekulé structures of B where $B - B[\mathcal{C}]$ is the sub-graph induced by the removal of the vertices belonging to $\mathcal{C}$ and its interior.*

Let us consider the benzenoid B described in Figure 22(b) and the cycle $\mathcal{C}$ depicted in red thick line. So, $B[\mathcal{C}]$ corresponds exactly to all the hexagons in the interior of $\mathcal{C}$, namely the hexagons depicted in red thick line in the middle figure. To compute the number of occurrences of $\mathcal{C}$ as a h-MCC, we have to compute the number of perfect matchings of the sub-graph induced by $B - B[\mathcal{C}]$. In this example, this sub-graph (depicted in red thick solid line in the rightmost

Algorithm 1: *Compute_Resonance_Energy (CRE-LF)*

Input: a benzenoid B
Output: the resonance energy $E(B)$

$energy \leftarrow 0$
foreach $K \in \mathcal{K}(B)$ **do**
  foreach $h = (v_1, v_2, \ldots, v_6) \in hexagons(B)$ **do**
    $config \leftarrow double_bonds_configuration(K, h)$
    if $config = 1$ **then**
      $P \leftarrow min_path(h, v_5, v_6)$
      if $straight_path(P)$ **then**
        $\mathcal{C} \leftarrow v_6 - v_1 - v_2 - v_3 - v_4 - v_5 \oplus min_path(h, v_5, v_6)$
      else $\mathcal{C} \leftarrow v_6 - v_5 \oplus min_path(h, v_5, v_6)$
    else if $config = 2$ **then**
      $\mathcal{C} \leftarrow v_1 - v_2 - v_3 - v_4 \oplus min_path(h, v_4, v_1)$
    else if $config = 3$ **then**
      $\mathcal{C}_1 \leftarrow v_1 - v_2 \oplus min_path(h, v_2, v_5) \oplus v_5 - v_6 \oplus min_path(h, v_6, v_1)$
      $\mathcal{C}_2 \leftarrow min_path(h, v_1 - v_2) \oplus v_2 - v_3 - v_4 - v_5 \oplus min_path(h, v_5, v_6)$
      $\mathcal{C} \leftarrow min_size(C_1, C_2)$
    else if $config = 4$ **then**
      $\mathcal{C}_1 \leftarrow min_path(h, v_1, v_2) \oplus v_2 - v_3 \oplus min_path(h, v_3, v_4) \oplus v_4 - v_5 \oplus min_path(h, v_5, v_6) \oplus v_6 - v_1$
      $\mathcal{C}_2 \leftarrow v_1 - v_2 \oplus min_path(h, v_2, v_3) \oplus v_3 - v_4 \oplus min_path(h, v_4, v_5) \oplus v_5 - v_6 \oplus min_path(h, v_6, v_1)$
      $\mathcal{C} \leftarrow min_size(C_1, C_2)$
    $energy \leftarrow energy + R_{|\mathcal{C}|}$
return $\frac{energy}{|\mathcal{K}(B)|}$

figure) has two perfect matchings. Moreover, we have $M(\mathcal{C}) = 1$ because if we consider the two Kekulé structures of $\mathcal{C}$ which allow it to be a conjugated circuit, there is only one of them for which $\mathcal{C}$ is a minimal circuit for at least one of its hexagon. So, we can conclude that $\mathcal{C}$ appears twice as an h-MCC in all the Kekulé structures of B.

To sum up, the method presented by Lin [Lin, 2000] (described in Algorithm 2) takes as input a benzenoid B and a base containing all the cycles of size at most 4 which can induce at least one h-MCC, and another base containing all the redundant circuits of the same sizes. In Line 1, it generates the set of cycles of B which belongs to the first base (we denote this set $\mathcal{C}^*$). Then, for each cycle in $\mathcal{C}^*$, it counts how many h-MCC are induced by this cycle in all the Kekulé structures of B (Lines 3-4), as shown in Figure 22(b). Then, it needs to find all couple of cycles of B which can produce one of the redundant circuits described in the second base and to ensure that the cycle having the largest size is not counted (Lines 5-8).

The main interest of this method is that it does not require to enumerate all the Kekulé structures of the given benzenoid. The only problem it has to solve is counting the number of perfect matchings in a graph. In the general case, counting the number of perfect matchings in a graph is a #P-complete problem. It is the same when the graph is bipartite. Fortunately, this task is proved to be polynomial for planar graphs [Kasteleyn, 1967] and an efficient method has been proposed for benzenoids [Rispoli, 2001].

7 Computing the Local Resonance Energy Thanks to CP

In this section, we describe how we implement the method of Lin and Fan and refine the method of Lin with the help of CP.

7.1 Implementing the Method of Lin and Fan Using CP

Lin and Fan have developed a software based on their method [Lin and Fan, 1999]. Unfortunately, this implementation is not available. As a consequence, in this part, we describe our implementation of their method based on constraint programming. As mentioned previously, this method cannot be efficient when the size of the considered benzenoids increases, due to the enumeration of all the Kekulé structures since the number of Kekulé structures may be exponential. Here, we mainly implement this method in order to assess the relevance of the approximation of the resonance energy made by the method described in Subsection 7.2.

Our implementation (denoted LFCP) exploits CP in order to enumerate all the Kekulé structures of a benzenoid

Algorithm 2: *Approximate_Resonance_Energy*

Input: a benzenoid B, a base of h-MCC, a base of redundant circuits
Output: the resonance energy $E(B)$

```
1 C* ← generate_circuits(B, 1, 4)
2 energy ← 0
3 foreach C ∈ C* do
4     energy ← energy + R_|C| × |K(B − B[C])| × M(C)
5 foreach (C1, C2) ∈ C* × C* do
6     if redundant(C1, C2) then
7         size ← max(|C1|, |C2|)
8         energy ← energy − R_size × |K(B − B[C1 ∪ C2])|
9 return energy / |K(B)|
```

(Line 2 of Algorithm 1). To do so, we model this problem as a CSP instance $P_1 = (X_1, D_1, C_1)$ for which every solution corresponds to a Kekulé structure. As any benzenoid $B = (V, E)$ is a bipartite graph, we can divide V into two disjoint sets V_1 and V_2 such that every edge of E links a vertex of V_1 to one of V_2. We consider a variable y_v per vertex v of V_1 whose domain contains every vertex w from V_2 such that $\{v, w\} \in E$. By so doing, if the variable y_v is assigned with value w, it means that the edge $\{v, w\}$ corresponds to a double bond. By definition of a solution, this ensures that there is a single double bond for any carbon atom of V_1. It remains to ensure the same property for the vertices of V_2. This can be achieved by considering an `all-different` constraint involving all the variables of X_1. So we obtain the following instance P_1:

$$\begin{cases} X_1 = \{y_v | v \in V_1\} \\ D_1 = \{D_{y_v} | v \in V_1\} \text{ with } D_{y_v} = \{w | w \in V_2,\ \{v, w\} \in E\} \\ C_1 = \{\texttt{all-different}(X_1)\} \end{cases}$$

Clearly, the solutions of P_1 correspond to the Kekulé structures of B and so to perfect matchings of B. Regarding the filtering of the `all-different` constraint, Regin proposed an efficient algorithm based on the matchings of a particular graph, called the *value graph* [Régin, 1994]. Note that, for our instance P_1, the value graph related to the `all-different` constraint we use is exactly the graph B. Moreover, any solver enforcing this filtering at each step of the search is able to enumerate efficiently the Kekulé structures since only assignments leading to solutions are explored. Note that another model was proposed [Mann and Thiel, 2013]. It considers binary variables and sum global constraints. In contrast, it does not provide any theoretical guarantee about the efficiency, unlike the model we propose.

7.2 Approximating the Local Resonance Energy

In this part, we propose a new method, using constraint programming, which refines the method proposed by Lin [Lin, 2000] in order to compute local aromaticity. Remember that local aromaticity is more useful than the global one since it helps to predict the parts of molecules where chemical reactions may take place while leading to global information like global aromaticity. Before going into details, we have to introduce some definitions.

7.2.1 Preliminary Definitions

First, we need to handle coordinates:

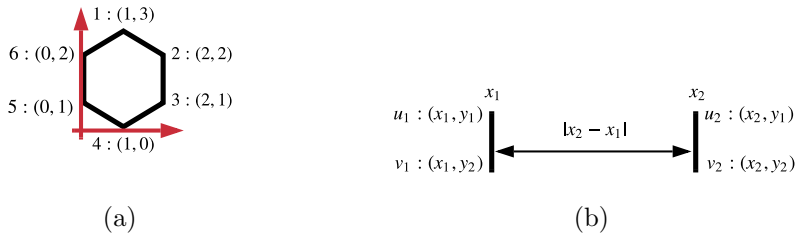
Definition 5 *Let $B = (V, E)$ be a benzenoid. A **coordinate function** $c : V \to \mathbb{Z}^2$ of B is a function that maps a couple of integers $(c(v).x, c(v).y)$ (i.e. an abscissa and an ordinate in the Cartesian coordinate plane) to each vertex v of B such that if $(v_1, v_2, v_3, v_4, v_5, v_6)$ are the vertices forming a hexagon (given clockwise) with v_1 the vertex having the largest ordinate, we have:*

$$\begin{cases} c(v_2) = (c(v_1).x + 1, c(v_1).y - 1) \\ c(v_3) = (c(v_1).x + 1, c(v_1).y - 2) \\ c(v_4) = (c(v_1).x, c(v_1).y - 3) \\ c(v_5) = (c(v_1).x - 1, c(v_1).y - 2) \\ c(v_6) = (c(v_1).x - 1, c(v_1).y - 1) \end{cases}$$

Figure 23(a) describes a simple example of coordinates for benzene.

Then, we consider some particular edges:

Definition 6 *Let B be a benzenoid and c a coordinate function. An edge $e = \{u, v\} \in E$ is a **vertical edge** of B if and only if $c(u).x = c(v).x$.*



(a) (b)

Figure 23: Benzene with coordinates (a) and example of interval (b).

The vertical edges of the benzenoid depicted in Figure 23(a) are $\{v_2, v_3\}$ and $\{v_5, v_6\}$. We now introduce the notion of interval related to vertical edges:

Definition 7 *Let B be a benzenoid and c a coordinate function. An **interval** I of B is a couple $I = (e_1, e_2)$ of vertical edges such as:*

$$\begin{cases} e_1 = \{u_1, v_1\} \in E \\ e_2 = \{u_2, v_2\} \in E \\ c(u_1).y = c(u_2).y \\ c(v_1).y = c(v_2).y \end{cases}$$

We denote:

$$\begin{cases} I.x_1 = c(u_1).x \\ I.y_1 = c(u_1).y \\ I.x_2 = c(u_2).x \\ I.y_2 = c(v_1).y \end{cases}$$

*We denote $|I| = |I.x_2 - I.x_1|$ the **size** of I.*

To sum up, an interval represents the space contained between two vertical edges that have the same ordinates. Figure 23(b) shows an example of interval.

7.2.2 Algorithm Description

We now describe our refined algorithm (called CRECP and described in Algorithm 3) based on constraint programming. This method takes as inputs a benzenoid $B = (V, E)$, a coordinate function c, a base containing all the cycles of size at most 4 that can induce at least one h-MCC, and another one containing all the couples of cycles of the first base which can form redundant circuits and it returns an approximation of the local energy $E(B, h)$ for each hexagon h. With this aim in view, we first compute the set $\mathcal{C}^*$ of all the cycles of B whose size is at most 6 (Line 1). Then, for each cycle $\mathcal{C}$ of $\mathcal{C}^*$ (Line 2), we first identify the cycle by a collection of intervals (Line 3) and, if $\mathcal{C}$ appears in the base of minimal circuits (Line 4), we enumerate all of the minimal circuits it induces and for each of them (Line 5), we add its contribution to the local resonance energy of the hexagon for which it is minimal (Lines 6-7). Note that for each cycle $\mathcal{C}$, we separately treat each of its minimal circuits, so we do not have to consider $M(\mathcal{C})$ anymore. However, $\mathcal{C}$ can also correspond to the contouring of the union of two redundant circuits over a hexagon h (Lines 8-9). If so, we do not have to take the contribution of the largest circuit into account (Lines 10-11). Finally, we divide the contribution of each hexagon by the number of Kekulé structures of B (Line 12). The main steps of Algorithm 3 are detailed below.

7.2.3 Enumeration of All the Cycles

We need to identify all the cycles which correspond to either a h-MCC of size at most 4 (Line 4 of Algorithm 3) or a union of two h-MCCs (Line 6). As the union of two h-MCCs of size 4 is at most of size 6, we have to enumerate all the cycles of size at most 6.

In order to enumerate all the cycles of size at most 6, we model this problem as a CSP instance $P_2 = (X_2, D_2, C_2)$. First, we consider a graph variable x_G whose domain is all the possible graphs between the empty graph and the graph B. This variable models the cycle we look for. To ensure that the value of this variable is a cycle, we impose the graph constraint `cycle` [Fages, 2014] on x_G. It remains to be ensured that the size of this cycle is at most 6. To this end, we introduce a Boolean variable x_e per edge e of B. x_e is set to 1 if the edge e appears in the graph depicted by x_G, 0 otherwise. Then, we use a collection of channeling constraints in order to link the variables x_e and the variable x_G. More precisely, for each edge e, we use a channeling constraint between x_e and x_G which imposes $x_e = 1 \iff e \textit{ appears in } x_G$. Finally, we add a global constraint `sum` over all the variables x_e to impose $\sum_{x_e | e \in E} x_e \in \{6, 10, 14, 18, 22, 26\}$ because we

Algorithm 3: *Compute_Resonance_Energy_CP(CRECP)*

Input: a benzenoid B, a coordinate function c, a base of h-MCC, a base of redundant circuits
Output: the local resonance energy $E(B,h)$ for each hexagon h of B

1 $\mathcal{C}^* \leftarrow generate_cycles_choco(B,1,6)$
2 **foreach** $\mathcal{C} \in \mathcal{C}^*$ **do**
3 $\quad id \leftarrow identify_cycle(\mathcal{C})$
4 $\quad$ **if** $in_minimal_circuits_base(id)$ **then**
5 $\quad\quad$ **foreach** $\mathcal{C}_m \in minimal_circuits(\mathcal{C})$ **do**
6 $\quad\quad\quad h \leftarrow$ hexagon s.t $\mathcal{C}_m$ is a h-MCC
7 $\quad\quad\quad energy[h] \leftarrow energy[h] + R_{|\mathcal{C}_m|} \times |\mathcal{K}(B - B[\mathcal{C}_m])|$
8 $\quad$ **else if** $in_redudant_circuits_base(id)$ **then**
9 $\quad\quad$ **foreach** $(\mathcal{C}_1, \mathcal{C}_2)$ $s.t.$ $\mathcal{C} = \mathcal{C}_1 \bigcup \mathcal{C}_2$ and $redundant(\mathcal{C}_1, \mathcal{C}_2, h)$ **do**
10 $\quad\quad\quad \mathcal{C}' \leftarrow circuit_with_max_size(\mathcal{C}_1, \mathcal{C}_2)$
11 $\quad\quad\quad energy[h] \leftarrow energy[h] - |\mathcal{K}(B - B[\mathcal{C}])| \times R_{|\mathcal{C}'|}$
12 **foreach** h of B **do** $energy[h] \leftarrow energy[h]/|\mathcal{K}(B)|$
13 **return** $energy$

consider circuits of size at most 6 and a circuit of size i has $4i+2$ edges. The channeling and sum constraints ensure that the size of the built cycle is suitable. At the end, we obtain the following instance P_2:

$$\begin{cases} X_2 = \{x_G\} \cup \{x_e | e \in E\} \\ D_2 = \{D_{x_G}\} \cup \{D_{x_e} | e \in E\} \text{ with } D_{x_G} = \{g | \emptyset \subseteq g \subseteq B\} \text{ and } D_{x_e} = \{0,1\} \\ C_2 = \{cycle(x_G), \sum_{x_e | e \in E} x_e \in \{6,10,14,18,22,26\}\} \cup \{channeling(x_e, x_G) | e \in E\} \end{cases}$$

As Choco implements graph variables and offers a large amount of graph-related constraints and global constraints, this model can be easily expressed with Choco.

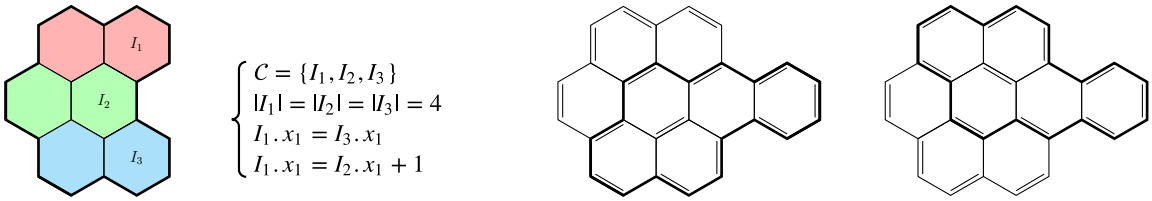
7.2.4 Counting the Number of Kekulé Structures

We need to count the number of Kekulé structures of B (Line 12) or $B - B[\mathcal{C}]$ for some cycles $\mathcal{C}$ of $\mathcal{C}^*$ (Lines 7 and 11). In [Rispoli, 2001], Rispoli presented a method that computes the number of Kekulé structures of a benzenoid. His method consists in transforming the given benzenoid $B = (V, E)$ into a specific matrix whose determinant is the number of Kekulé structures of B. Formally, it relies again on the property that V can be divided into two disjoint sets V_1 and V_2 such that every edge of E links a vertex of V_1 to one of V_2. It computes the *biadjacency matrix* of B whose rows (resp. columns) are labeled by the vertices of V_1 (resp. V_2) and such that the value for row $v_1 \in V_1$ and column $v_2 \in V_2$ is 1 if the edge $\{v_1, v_2\} \in E$, 0 otherwise. Rispoli showed that the determinant of the biadjacency matrix of B corresponds to the number of Kekulé structures of B assuming that $|V_1| = |V_2|$[6]. Interestingly, this computation can be achieved in polynomial time. So we exploit it in order to efficiently compute the number of Kekulé structures of B (Line 12).

Regarding Lines 7 and 12, we consider the number of Kekulé structures of $B - B[\mathcal{C}]$. $B - B[\mathcal{C}]$ is the part of the benzenoid B (i.e. a sub-graph) induced by the removal of the vertices belonging to $\mathcal{C}$ (including its interior). In practice, this sub-graph does not necessarily correspond to a benzenoid and so the method of Rispoli cannot be applied directly. For instance, in Figure 22(b), the corresponding sub-graph has two connected components. Reasoning in terms of connected components is not enough since, in this figure, the bottom component is not a benzenoid unlike the top one. A solution consists in removing all vertices having a degree of one and their neighbors until no such vertices remain. Indeed, by construction, a vertex having a degree of one is necessarily connected by a double bond to its neighbor and so has no impact on the number of Kekulé structures. At the end of the process, the remaining vertices have a degree of zero or at least two. In the first case, the connected component contains a single vertex and has no Kekulé structure. In contrast, in the second case, the connected component has several vertices. Moreover, every vertex belongs to cycles whose length is $4i+2$ where i is an integer. This latter condition is important since we can easily show that the result of Rispoli can be extended to such graphs. Indeed, the proof of this result relies on these cycles of length $4i+2$. So the method of Rispoli can be used for counting the number of Kekulé structures of B (Line 12) or $B - B[\mathcal{C}]$[7].

[6] If $|V_1| \neq |V_2|$, it is trivial to show that the benzenoid has no Kekulé structure.

[7] In [Carissan et al., 2020a], this step is achieved by enumerating all the Kekulé structures thanks to the CSP model P_1. This choice was justified by the long computation time of the determinant as well as by the fact that $B - B[\mathcal{C}]$ was not a benzenoid. The extension of Rispoli's result and the use of a more efficient library for the computation of the determinant allow us to make a different choice here. Note that we compared experimentally (not reported here) the determinant calculation, the solution counting for the model P_1 or one of Mann and Thiel [Mann and Thiel, 2013], and not surprisingly, the first method is the most efficient.



(a) (b)

Figure 24: A cycle and the relations between its intervals (a), and a second example of redundant circuits (b).

7.2.5 Identification of Cycles

Once Choco returns a cycle, we need to determine if this cycle belongs to the base of h-MCCs (Line 4) or one of the redundant circuits (Line 6). For achieving this task, we first represent any cycle by the bias of a set of intervals and some relations between these intervals. The purpose of these relations is to represent the distance between two vertical edges of each couple of intervals (either the two left edges or the two right edges). So, given a set of intervals with this kind of relations, we are able to build the associated cycle and vice versa. Figures 24(a) shows an example of such a representation. Accordingly, we construct each base by describing every cycle (h-MCC or redundant cycle) identified by Lin [Lin, 2000] by a set of intervals and some relations between them. Now, each time a new cycle is returned by Choco, we translate it into a set of intervals and their relations, and check if it belongs to one of the two bases by simply comparing sets and relations. So, we are able to determine if this cycle can induce h-MCC or if it can be obtained by the union of redundant circuits.

Furthermore, the second base contains, for each cycle $\mathcal{C}$, a set of a couple of cycles whose union forms $\mathcal{C}$. This allows us to remove the energy that we have over-counted due to redundant circuits (Line 10). For example, let us consider $\mathcal{C}$ as being the union of the two circuits represented in Figure 24(b). It can be obtained either by the union of the cycles of the leftmost figure (of sizes 3 and 4), or by the cycles of the rightmost one (of sizes 3 and 4 too). So, each time Choco finds $\mathcal{C}$ in a benzenoid, we need to remove the energy associated with two circuits of size 4.

8 Computing the Local Resonance Energy in Practice

This section is devoted to an experimental comparison of three methods able to estimate the aromaticity of benzenoids. More precisely, we assess the behavior of the two methods we define, namely CRECP and LFCP, and the reference method NICS. CRECP and LFCP consist of accounting for the energy attached to certain circuits thanks to CP techniques while NICS relies on quantum calculations. We first describe our experimental protocol. Then, as CRECP only takes into account a part of the circuits considered by LFCP, we compare empirically, these two methods in order to assess the relevance of this approximation. Finally, CRECP is compared with NICS.

8.1 Experimental Protocol

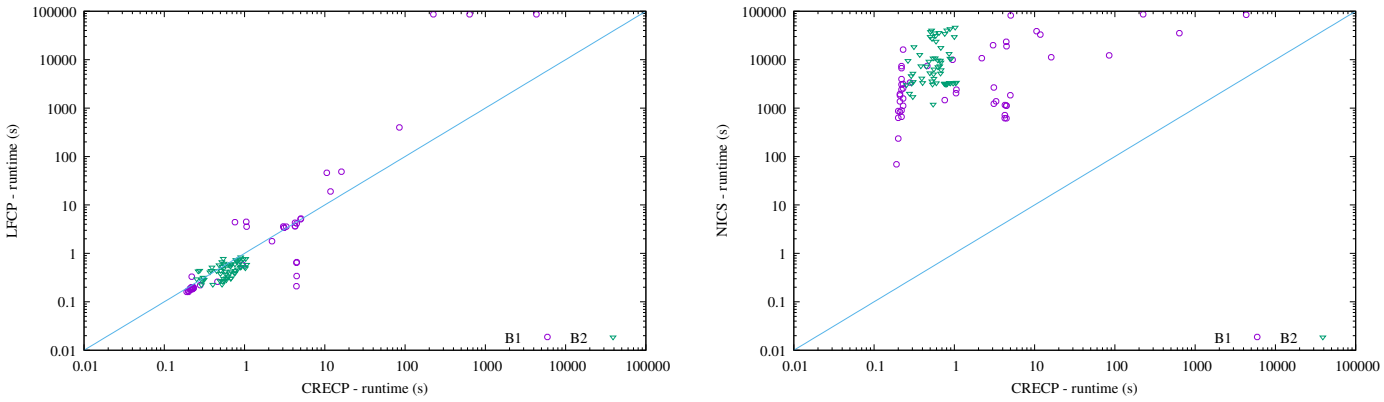
The methods CRECP and LFCP are implemented in Java and compiled with Java SE 1.8. They can be found as a part of our BenzAI software[8]. Their implementation relies on Choco Solver (v. 4.10.7) for which we used the default settings of Choco. Note that, according to [Lin, 2000], the method proposed by Lin has already been implemented. Unfortunately, it does not seem to be available. We use $R_i = \frac{1}{i^2}$ instead of the optimized values because the optimized values are only defined for circuits of size at most 4, which is not sufficient for LFCP. However, using the optimized values (or not) does not influence the comparisons of CRECP and NICS. For NICS, we exploit the implementation provided in a commercial program (see `http://gaussian.com/`). Remember that NICS is the reference method for chemists when they want to estimate aromaticity. All the methods are run on servers with 2.20 GHz Intel Xeon Gold processor and 256 Gb under CentOS Linux release 8.1.1911. We limit the runtime to 24 hours.

Now, we describe the set of molecules we consider for our experiments. This set is divided into three subsets:

- $\mathcal{B}_1$ is a set of 48 benzenoids whose size varies from one hexagon to 109. It is composed of several molecules which are well known to chemists and often used in such comparisons (e.g. [Randić, 2019, Ruiz-Morales, 2004]). It corresponds to the set used in [Carissan et al., 2020a, Randić, 2019] augmented by the molecules used in [Ruiz-Morales, 2004]. All the considered molecules are depicted in Appendix (see Figures 28-33).

- $\mathcal{B}_2$ is a selection of 50 catacondensed benzenoids having from 5 to 10 hexagons. For each considered value of n, we randomly select 10 molecules among all the catacondensed benzenoids having n hexagons. Catacondensed benzenoids are well known to be benzenoids having a lot of circuits of small sizes.

[8] Available at `https://benzai-team.github.io/BenzAI/`.



(a) (b)

Figure 25: Comparison of runtimes on sets $\mathcal{B}_1$ and $\mathcal{B}_2$: CRECP vs LFCP (a)and CRCEP vs NICS (b).

- $\mathcal{B}_3$ contains rectangular benzenoids. We fix their height to 5 hexagons and vary their width from 1 to 20. Rectangular benzenoids are benzenoids having circuits of various sizes and, in particular, circuits of large sizes. Note that we also achieved experiments by varying the height. However, they led to similar trends. So we do not report them here.

The comparisons rely on two criteria: the runtime and the quality of the estimation. For the runtime, we consider the CPU time required for the run of each method. The second criterion is more difficult to assess and is an important question from a chemical viewpoint. First, remember that aromaticity cannot be measured. So, there is no exact value to which we can refer in order to estimate the quality of our computation. The methods CRECP and LFCP are quite different from NICS. In particular, the values they return cannot lead to similar numbers for NICS, even if all these approaches describe local aromaticity. More precisely, CRECP and LFCP give values between 0 and 1, whereas, for NICS, the values are not bound to a specific interval. Indeed, CRECP and LFCP aim to describe the behavior of the electronic structure of the molecule as a superposition of closed electronic circuits whereas the NICS approach measures how much the electronic structure would be distorted by an external magnetic field. However, their trends should coincide. Hence, whatever the two considered methods M_1 and M_2, we compare them by computing the sample Pearson correlation coefficient from the values they produce. More precisely, given a benzenoid B, we apply M_1 and M_2 on B. In return, we obtain two values $v_h^{M_1}$ and $v_h^{M_2}$ per hexagon h which assess the aromaticity of h according to methods M_1 and M_2 respectively. The sample Pearson correlation coefficient for the benzenoid B is then computed thanks to the following formula:

$$r_{M_1,M_2} = \frac{n\sum_h v_h^{M_1} v_h^{M_2} - \sum_h v_h^{M_1} \sum_h v_h^{M_2}}{\sqrt{n\sum_h v_h^{M_1} - \left(\sum_h v_h^{M_1}\right)^2}\sqrt{n\sum_h v_h^{M_2} - \left(\sum_h v_h^{M_2}\right)^2}}$$

where n is the number of hexagons of B.

8.2 Comparing CRECP with LFCP

We first compare CRECP and LFCP with respect to the runtime. Regarding the set $\mathcal{B}_1$ (see Table 9 and Figure 25(a)), the two methods obtain similar results for most of the molecules even if we can note that CRECP seems to be slower for the more compact ones. Indeed, by considering the set $\mathcal{B}_2$, we can see that both methods require a similar runtime, which is foreseeable since catacondensed benzenoids have a tree structure and so are far from being compact molecules. In contrast, for rectangle benzenoids which are by definition compact molecules, CRECP seems to be slower than LFCP (see Figure 26). However, this holds until a given size above which CRECP turns out to be significantly faster than LFCP. A possible explanation is related to the number of Kekulé structures. This number may grow exponentially with the number of hexagons. At some point, enumerating all the Kekulé structures becomes too time-expensive. For instance, we can see that LFCP reaches the timeout (TO) for the three benzenoids in Table 9 that have the largest number of Kekulé structures. At the same time, the number of circuits grows but more slowly and so CRECP is still usable. For instance, LFCP requires about 23,300 s to process the rectangle benzenoid of dimensions 5×14 while CRECP treats one of the dimensions 5×16 in a similar runtime (22,516 s). Moreover, in our experiments, CRECP processes each considered benzenoid within the time limit.

Now, regarding the quality of the estimation of the aromaticity, we compute the sample Pearson correlation coefficient between CRECP and LFCP. Whatever the benzenoid belonging to sets $\mathcal{B}_1$, $\mathcal{B}_2$ or $\mathcal{B}_3$, the coefficient has a value between

Id	n	# Kekulé structures	Runtime			coef.
			CRECP	LFCP	NICS	
1	1	2	0.19	0.16	69	1.00
2	2	3	0.20	0.16	235	1.00
3	3	5	0.20	0.17	872	1.00
4	3	4	0.20	0.17	631	1.00
5	4	5	0.21	0.18	831	1.00
6	4	8	0.21	0.18	1,377	1.00
7	4	9	0.21	0.19	1,827	1.00
8	4	6	0.22	0.18	661	1.00
9	4	7	0.21	0.18	1,950	0.79
10	5	6	0.22	0.33	894	1.00
11	5	9	0.22	0.19	3,069	0.55
12	5	10	0.22	0.19	2,415	0.99
13	5	12	0.22	0.20	3,979	0.93
14	5	11	0.22	0.20	6,684	0.73
15	5	13	0.22	0.20	7,351	0.99
16	5	13	0.23	0.20	2,557	0.85
17	5	14	0.23	0.20	16,148	0.96
18	5	11	0.23	0.19	1,580	0.90
19	5	9	0.23	0.19	3,137	0.92
20	5	9	0.23	0.19	1,116	1.00
21	7	20	0.28	0.22	3,373	1.00
22	13	250	0.95	0.58	9,968	0.94
23	20	3,250	4.47	4.15	18,929	0.93
24	19	3,100	3.04	3.60	19,854	0.90
25	24	16,100	11.78	18.97	33,035	0.93
26	25	34,560	10.57	46.10	38,727	0.84
27	11	25	0.46	0.26	7,425	0.97
28	17	1,320	2.20	1.78	10,722	0.93
29	8	34	4.47	0.66	1,121	0.87
30	8	31	4.43	0.64	23,420	0.96
31	8	19	4.43	0.21	1,124	0.97
32	8	16	4.46	0.34	615	1.00
33	9	40	1.06	3.57	2,408	0.94
34	9	20	3.10	3.38	1,229	0.99
35	9	30	3.10	3.40	2,662	0.99
36	10	50	3.31	3.56	1,383	0.99
37	14	175	4.98	5.06	1,842	0.99
38	8	45	0.76	4.40	1,460	0.96
39	10	101	1.05	4.48	2,034	0.96
40	6	14	4.26	4.29	716	0.96
41	6	10	4.24	3.64	618	0.99
42	6	14	4.27	3.65	1,155	0.91
43	13	432	5.04	5.23	81,822	0.94
44	61	267,227,532	635.58	TO	35,322	0.97
45	49	126,672,896	225.23	TO	TO	-
46	30	27,508	16.10	48.60	11,222	0.48
47	37	232,848	84.99	399.15	12,276	0.99
48	109	53,930,238,785,494,000	4,324.50	TO	84,396	0.91

Table 9: Number of hexagons, Runtime (in seconds) of CRECP, LFCP and NICS, and value of the Pearson correlation coefficient between CRECP and NICS for the molecules of $\mathcal{B}_1$.

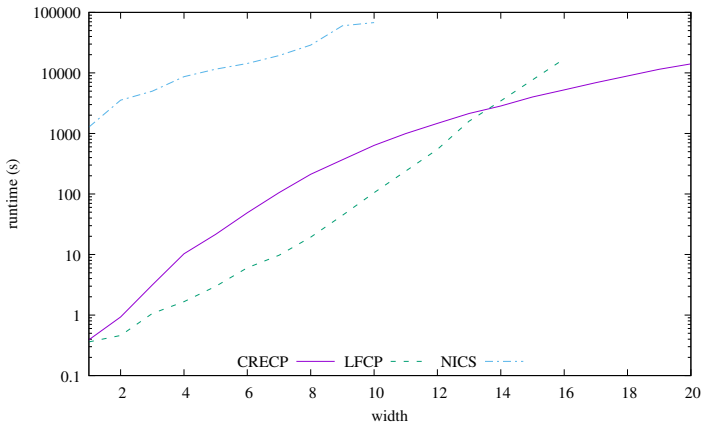


Figure 26: Runtime (in seconds) of CRECP, LFCP and NICS for rectangle benzenoids whose height is fixed to five and whose width varies from 1 to 20.

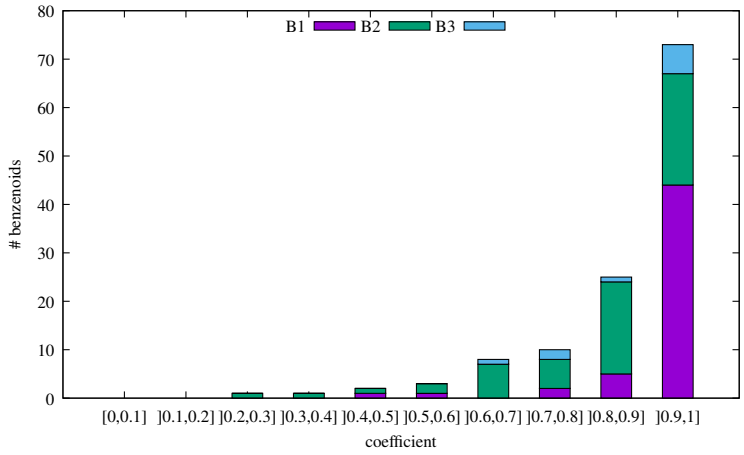


Figure 27: The sample Pearson correlation coefficient between CRECP and NICS for sets $\mathcal{B}_1$, $\mathcal{B}_2$ and $\mathcal{B}_3$.

0.9997 and 1. In other words, the linear relationship is almost perfect. We can therefore consider the two methods as equivalent with respect to this criterion. Such a result was foreseeable. Indeed, both methods rely on the same idea by evaluating the resonance energy of each hexagon. The main difference is that LFCP considers circuits of any size while CRECP only takes into account circuits of size at most 4. Since the contribution of circuits to the resonance energy decreases with their size, disregarding circuits of size greater than 4 has little influence on the final result.

8.3 Comparing CRECP with NICS

As CRECP and LFCP obtain similar results, we only compare CRECP with NICS. Clearly, the runtime is in favor of CRECP which turns out to be more efficient with several orders of magnitude whatever the benzenoid and the benchmark we consider (see Figures 25(b) and 26). For example, CRECP processes molecule 26 of $\mathcal{B}_1$ in 10.57 s against 38,727 s for NICS. For some benzenoid structures (e.g. the largest considered rectangular benzenoids), NICS exceeds the time limit and may require several days of computations. In some way, this result was foreseeable since NICS requires complex quantum calculations and such calculations are very time-consuming.

Regarding the quality of the estimation of the aromaticity, we consider the sample Pearson correlation coefficient between CRECP and NICS (see Figure 27 and Table 9). Figure 27 provides the number of benzenoids for various intervals of values of the coefficient for each considered set. Globally, we can note that for about 78% of the considered benzenoids, the coefficient is greater than 0.8. Better still, for 57%, the coefficient is greater than 0.9. The coefficient is less than 0.5 for only 3% of the considered benzenoids. Looking more closely at the results, we can note that the value of the coefficient is not related to the size of the benzenoid. However, the shape may have some influence as shown in Figure 27. In the end, it turns out that the CRECP and NICS trends coincide most of the time. It follows that chemists can exploit CRECP to assess the aromaticity in a faster way than with NICS while having a reliable estimate.

9 Conclusions and Perspectives

In this paper, we addressed two important issues in theoretical chemistry. On one hand, we considered the exhaustive generation of benzenoid structures satisfying a certain amount of properties. In this context, we proposed an approach based on constraint programming. Its main advantage w.r.t. existing methods in the literature lies in its flexibility. Indeed, we can take into account the wishes of chemists by simply adding variables and/or constraints to our general model while existing bespoke methods rely on more rigid and complex notions and cannot be adapted without requiring heavy tasks. Moreover, our approach turns out to be more general, making it possible to generate benzenoids with holes for instance. On the other hand, we tackled the problem of estimating the aromaticity, which is a fundamental concept in chemistry. We mainly presented a new method based on constraint programming for computing the local aromaticity of benzenoids. This method refines the method proposed by Lin by dealing with local aromaticity instead of global one. In practice, it provides a reliable estimate by mainly giving the same trends as NICS (which is considered as a reference by theoretical chemists) while turning out to be significantly faster than NICS by several orders of magnitude.

Chemists are interested in exhaustively generating benzenoid structures with particular shapes (e.g. [Trinquier and Malrieu, 2018, Bauschlicher et al., 2018]). We have already dealt with the main shapes in this paper. So a natural extension of this work relies on taking into account other specific properties related to the needs of chemists. Among these properties, a hot topic in theoretical chemistry is related to local properties which can be described as patterns (e.g. [Qiu et al., 2020, Liu and Feng, 2020, Ajayakumar et al., 2021, Chen et al., 2021, Cheung et al., 2021, Fujise et al., 2021, Kancherla and Jørgensen, 2020, Uryu et al., 2020, Xia et al., 2021]). We have already started to explore this promising avenue [Carissan et al., 2021]. Another step consists in studying the limit of our approach both in terms of properties we can express and our ability to exhaustively generate benzenoids of large size. In addition, we need to look at the impact from a chemistry perspective. Our exhaustive generation can be useful to complete some databases (e.g. [Bauschlicher et al., 2018]). By combining our two contributions, we offer chemists the opportunity to study aromaticity on much larger collections of benzenoids than ever before. Beyond these contributions, this paper shows how, once again, constraint programming can be useful to tackle and solve problems related to theoretical chemistry [Mann et al., 2014, Wu, 2004, Simoncini et al., 2015, Ismail et al., 2019, Kim et al., 2018]. In particular, many questions about benzenoids can be modeled as decision or optimization problems under constraints (e.g. computing the Clar number or finding the closest structure to a Kekulé structure when no Kekulé structure exists) and can correspond to difficult tasks (e.g. computing the Clar number is NP-hard [Bérczi-Kovács and Bernáth, 2018]). It could be of interest for both communities to study them.

10 Acknowledgements

The authors would like to thank Mohamed Sami Cherif and the anonymous reviewers for their useful comments.

References

M. R. Ajayakumar, Ji Ma, Andrea Lucotti, Karl Sebastian Schellhammer, Gianluca Serra, Evgenia Dmitrieva, Marco Rosenkranz, Hartmut Komber, Junzhi Liu, Frank Ortmann, Matteo Tommasini, and Xinliang Feng. Persistent peri-Heptacene: Synthesis and In Situ Characterization. *Angew. Chem. Int. Ed.*, 2021. ISSN 1521-3773. doi: 10.1002/anie.202102757.

L. J. Allamandola, D. M. Hudgins, and S. A. Sandford. Modeling the Unidentified Infrared Emission with Combinations of Polycyclic Aromatic Hydrocarbons. *The Astrophysical Journal*, 511(2):L115–L119, 1999. doi: 10.1086/311843.

Cyril Aumaitre and Jean-François Morin. Polycyclic Aromatic Hydrocarbons as Potential Building Blocks for Organic Solar Cells. *The Chemical Record*, 19(6):1142–1154, 2019. doi: 10.1002/tcr.201900016. URL https://onlinelibrary.wiley.com/doi/abs/10.1002/tcr.201900016.

Charles W. Bauschlicher, A. Ricca, C. Boersma, and L. J. Allamandola. The NASA ames PAH IR spectroscopic database: Computational version 3.00 with updated content and the introduction of multiple scaling factors. *The Astrophysical Journal Supplement Series*, 234(2):32, feb 2018. doi: 10.3847/1538-4365/aaa019. URL https://doi.org/10.3847%2F1538-4365%2Faaa019.

Charles W. Bauschlicher, Jr., Els Peeters, and Louis J. Allamandola. The Infrared Spectra of Very Large, Compact, Highly Symmetric, Polycyclic Aromatic Hydrocarbons (PAHs). *The Astrophysical Journal*, 678(1):316–327, 2008. doi: 10.1086/533424.

Uliana Beser, Marcel Kastler, Ali Maghsoumi, Manfred Wagner, Chiara Castiglioni, Matteo Tommasini, Akimitsu Narita, Xinliang Feng, and Klaus Müllen. A C216-Nanographene Molecule with Defined Cavity as Extended Coronoid. *Journal of the American Chemical Society*, 138(13):4322–4325, 2016. doi: 10.1021/jacs.6b01181.

Jordy Bouwman, Harold Linnartz, and Alexander G.G.M. Tielens. Mid-infrared spectroscopic signatures of dibenzopyrene cations – the effect of symmetry on pah ir spectroscopy. *Journal of Molecular Spectroscopy*, 378:111458, 2021. ISSN 0022-2852. doi: https://doi.org/10.1016/j.jms.2021.111458. URL https://www.sciencedirect.com/science/article/pii/S0022285221000424.

Bouwman, J., Castellanos, P., Bulak, M., Terwisscha van Scheltinga, J., Cami, J., Linnartz, H., and Tielens, A. G. G. M. Effect of molecular structure on the infrared signatures of astronomically relevant pahs. *A&A*, 621:A80, 2019. doi: 10.1051/0004-6361/201834130. URL https://doi.org/10.1051/0004-6361/201834130.

G. Brinkmann, G. Caporossi, and P. Hansen. A Constructive Enumeration of Fusenes and Benzenoids. *Journal of Algorithms*, 45(2), 2002.

J. Brunvoll, R. N. Cyvin, and S. J. Cyvin. Enumeration and Classification of Double Coronoid Hydrocarbons – Appendix: Triple Coronoids. *Croatica Chemica Acta*, 63(4):585–601, 1990.

E.R. Bérczi-Kovács and A. Bernáth. The complexity of the Clar number problem and an exact algorithm. *J Math Chem*, 56:597–605, 2018. doi: 10.1007/s10910-017-0799-8.

Gilles Caporossi and Pierre Hansen. Enumeration of polyhex hydrocarbons to h = 21. *Journal of Chemical Information and Computer Sciences*, 38(4):610–619, 1998. doi: 10.1021/ci970116n. URL https://doi.org/10.1021/ci970116n.

Y. Carissan, C.-A. Dim, D. Hagebaum-Reignier, N. Prcovic, C. Terrioux, and A. Varet. Computing the Local Aromaticity of Benzenoids Thanks to Constraint Programming. In *CP*, pages 673–689, 2020a.

Y. Carissan, D. Hagebaum-Reignier, N. Prcovic, C. Terrioux, and A. Varet. Using Constraint Programming to Generate Benzenoid Structures in Theoretical Chemistry. In *CP*, pages 690–706, 2020b.

Y. Carissan, D. Hagebaum-Reignier, N. Prcovic, C. Terrioux, and A. Varet. Exhaustive Generation of Benzenoid Structures Sharing Common Patterns. In *CP*, pages 19:1–19:18, 2021.

Ying Chen, Chaojun Lin, Zhixing Luo, Zhibo Yin, Haonan Shi, Yanpeng Zhu, and Jiaobing Wang. Double π-Extended Undecabenzo[7]helicene. *Angew. Chem. Int. Ed.*, 60(14):7796–7801, 2021. ISSN 1521-3773. doi: 10.1002/anie.202014621.

Z. Chen, C. S. Wannere, C. Corminboeuf, R. Puchta, and P. von Ragué Schleyer. Nucleus-Independent Chemical Shifts (NICS) as an Aromaticity Criterion. *Chem Rev*, 105:3842–3888, 2005.

Kwan Yin Cheung, Kosuke Watanabe, Yasutomo Segawa, and Kenichiro Itami. Synthesis of a zigzag carbon nanobelt. *Nat. Chem.*, 13(3):255–259, March 2021. ISSN 1755-4349. doi: 10.1038/s41557-020-00627-5.

E. Clar. *The Aromatic Sextet*. Wiley, 1972.

E. Clar and R. Schoental. *Polycyclic Hydrocarbons Volume 1*. Springer Berlin, Berlin, 1964.

Caterina Cocchi, Deborah Prezzi, Alice Ruini, Marilia J. Caldas, and Elisa Molinari. Anisotropy and size effects on the optical spectra of polycyclic aromatic hydrocarbons. *The Journal of Physical Chemistry A*, 118(33):6507–6513, 2014. doi: 10.1021/jp503054j.

J Cyvin, J Brunvoll, and B N Cyvin. Search for Concealed Non-Kekuléan Benzenoids and Coronoids. *J. Chem. Inf. Comput. Sci.*, 29(4):237, 1989.

Jo Devriendt, Bart Bogaerts, Maurice Bruynooghe, and Marc Denecker. Improved static symmetry breaking for sat. In Nadia Creignou and Daniel Le Berre, editors, *Theory and Applications of Satisfiability Testing – SAT 2016*, pages 104–122, 2016.

Marco Di Giovannantonio, Xuelin Yao, Kristjan Eimre, José I. Urgel, Pascal Ruffieux, Carlo A. Pignedoli, Klaus Müllen, Roman Fasel, and Akimitsu Narita. Large-Cavity Coronoids with Different Inner and Outer Edge Structures. *Journal of the American Chemical Society*, 142(28):12046–12050, 2020. doi: 10.1021/jacs.0c05268.

J. R. Dias. Valence-bond determination of diradical character of polycyclic aromatic hydrocarbons: From acenes to rectangular benzenoids. *J. Phys. Chem. A*, 117:4716–4725, 2013.

Jerry Ray Dias. Structure and Electronic Characteristics of Coronoid Polycyclic Aromatic Hydrocarbons as Potential Models of Graphite Layers with Hole Defects. *The Journal of Physical Chemistry A*, 112(47):12281–12292, 2008. doi: 10.1021/jp806987f.

B. T. Draine. Astronomical Models of PAHs and Dust. *EAS Publications Series*, 46:29–42, 2011. doi: 10.1051/eas/1146003.

Philip E. Eaton and Thomas W. Cole. Cubane. *Journal of the American Chemical Society*, 86(15):3157–3158, 1964. doi: 10.1021/ja01069a041.

J.-G. Fages, X. Lorca, and C. Prud'homme. Choco solver user guide documentation. https://choco-solver.readthedocs.io/en/latest/.

Jean-Guillaume Fages. *Exploitation de structures de graphe en programmation par contraintes.* PhD thesis, École des mines de Nantes, France, 2014.

Kei Fujise, Eiji Tsurumaki, Kan Wakamatsu, and Shinji Toyota. Construction of Helical Structures with Multiple Fused Anthracenes: Structures and Properties of Long Expanded Helicenes. *Chemistry – A European Journal*, 27(14):4548–4552, March 2021. ISSN 0947-6539. doi: 10.1002/chem.202004720.

E. Hückel. Quantentheoretische Beiträge zum Benzolproblem. *Zeitschrift für Physik*, 70:204–286, 1931.

Idil Ismail, Holly B. V. A. Stuttaford-Fowler, Curtis Ochan Ashok, Christopher Robertson, and Scott Habershon. Automatic Proposal of Multistep Reaction Mechanisms using a Graph-Driven Search. *The Journal of Physical Chemistry A*, 123(15):3407–3417, 2019. doi: 10.1021/acs.jpca.9b01014.

Sindhu Kancherla and Kåre B. Jørgensen. Synthesis of Phenacene–Helicene Hybrids by Directed Remote Metalation. *J. Org. Chem.*, 85(17):11140–11153, September 2020. ISSN 0022-3263. doi: 10.1021/acs.joc.0c01097.

P. W. Kasteleyn. *Graph theory and crystal physics*, page 43–110. Academic Press, 1967.

Marcel Kastler, Jochen Schmidt, Wojciech Pisula, Daniel Sebastiani, and Klaus Müllen. From Armchair to Zigzag Peripheries in Nanographenes. *Journal of the American Chemical Society*, 128(29):9526–9534, 2006. doi: 10.1021/ja062026h.

Aug. Kekulé. Untersuchungen über aromatische verbindungen ueber die constitution der aromatischen verbindungen. *Justus Liebigs Annalen der Chemie*, 137(2):129–196, 1866. doi: 10.1002/jlac.18661370202. URL https://chemistry-europe.onlinelibrary.wiley.com/doi/abs/10.1002/jlac.18661370202.

Yeonjoon Kim, Jin Woo Kim, Zeehyo Kim, and Woo Youn Kim. Efficient prediction of reaction paths through molecular graph and reaction network analysis. *Chemical Science*, 9(4):825–835, 2018. doi: 10.1039/C7SC03628K.

Akihito Konishi, Koki Horii, Daisuke Shiomi, Kazunobu Sato, Takeji Takui, and Makoto Yasuda. Open-Shell and Antiaromatic Character Induced by the Highly Symmetric Geometry of the Planar Heptalene Structure: Synthesis and Characterization of a Nonalternant Isomer of Bisanthene. *Journal of the American Chemical Society*, 2019. doi: 10.1021/jacs.9b04080.

C. Lecoutre. *Constraint Networks: Techniques and Algorithms.* Wiley, 2009.

C. Lin. Efficient Method for Calculating the Resonance Energy Expression of Benzenoid Hydrocarbons Based on the Enumeration of Conjugated Circuits. *J. Chem. Inf. Comput. Sci.*, 40:778–783, 2000.

C. Lin and G. Fan. Algorithms for the Count of Linearly Independent and Minimal Conjugated Circuits in Benzenoid Hydrocarbons. *J. Chem. Inf. Comput. Sci*, 39:782–787, 1999.

Junzhi Liu and Xinliang Feng. Synthetic tailoring of graphene nanostructures with Zigzag-Edged topologies: Progress and perspectives. *Angewandte Chemie International Edition*, 59:2–18, 2020. doi: 10.1002/anie.202008838.

H.C. Longuet-Higgins. The symmetry groups of non-rigid molecules. *Molecular Physics*, 6(5):445–460, 1963. doi: 10.1080/00268976300100501.

A. Luch. *The Carcinogenic Effects of Polycyclic Aromatic Hydrocarbons.* Imperial College Press, London, 2005. URL https://www.worldscientific.com/worldscibooks/10.1142/p306.

M. Mann and B. Thiel. Kekulé Structures Enumeration Yields Unique SMILES. In *Proceedings of Workshop on Constraint Based Methods for Bioinformatics*, 2013.

Martin Mann, Feras Nahar, Norah Schnorr, Rolf Backofen, Peter F. Stadler, and Christoph Flamm. Atom mapping with constraint programming. *Algorithms for Molecular Biology*, 9(1):23, 2014. doi: 10.1186/s13015-014-0023-3.

Shantanu Mishra, Thorsten G. Lohr, Carlo A. Pignedoli, Junzhi Liu, Reinhard Berger, José I. Urgel, Klaus Müllen, Xinliang Feng, Pascal Ruffieux, and Roman Fasel. Tailoring Bond Topologies in Open-Shell Graphene Nanostructures. *ACS Nano*, 12(12):11917–11927, 2018. doi: 10.1021/acsnano.8b07225.

Shantanu Mishra, Doreen Beyer, Kristjan Eimre, Junzhi Liu, Reinhard Berger, Oliver Gröning, Carlo A. Pignedoli, Klaus Müllen, Roman Fasel, Xinliang Feng, and Pascal Ruffieux. Synthesis and Characterization of π-Extended Triangulene. *Journal of the American Chemical Society*, 141(27):10621–10625, 2019. doi: 10.1021/jacs.9b05319.

Shantanu Mishra, Doreen Beyer, Kristjan Eimre, Shawulienu Kezilebieke, Reinhard Berger, Oliver Gröning, Carlo A. Pignedoli, Klaus Müllen, Peter Liljeroth, Pascal Ruffieux, Xinliang Feng, and Roman Fasel. Topological frustration induces unconventional magnetism in a nanographene. *Nature Nanotechnology*, 15(1):22–28, 2020. doi: 10.1038/s41565-019-0577-9.

Akimitsu Narita, Xiao-Ye Wang, Xinliang Feng, and Klaus Müllen. New advances in nanographene chemistry. *Chemical Society Reviews*, 44(18):6616–6643, 2015. doi: 10.1039/C5CS00183H.

Zijie Qiu, Akimitsu Narita, and Klaus Müllen. Carbon nanostructures by macromolecular design from branched polyphenylenes to nanographenes and graphene nanoribbons. *Faraday Discussions*, 2020. doi: 10.1039/D0FD00023J. Publisher: The Royal Society of Chemistry.

M. Randić. Aromaticity of Polycyclic Conjugated Hydrocarbons. *Chemical Reviews*, 103(9):3449–3606, 2003. doi: 10.1021/cr9903656.

M. Randić. Benzenoid Rings Resonance Energies and Local Aromaticity of Benzenoid Hydrocarbons. *Journal of Computational Chemistry*, 40(5):753–762, 2019.

M. Randić, X. Guo, and D. J. Klein. Analytical Expressions for the Count of LM-Conjugated Circuits of Benzenoid Hydrocarbons. *Int. J. Quantum Chem.*, 60:943–958, 1996.

S. Rayne and K. Forest. Singlet-triplet ($S_0 \rightarrow T_1$) excitation energies of the $[4 \times n]$ rectangular graphene nanoribbon series (n=2-6): A comparative theoretical study. *Comput. Theor. Chem.*, 976:105–112, 2011.

Alessandra Ricca, Charles W. Bauschlicher, Christiaan Boersma, Alexander G. G. M. Tielens, and Louis J. Allamandola. The Infrared spectroscopy of compact polycyclic aromatic hydrocarbons containing up to 384 carbons. *The Astrophysical Journal*, 754(1):75, 2012. doi: 10.1088/0004-637X/754/1/75.

Alessandra Ricca, Joseph E. Roser, Els Peeters, and Christiaan Boersma. Polycyclic Aromatic Hydrocarbons with Armchair Edges: Potential Emitters in Class B Sources. *The Astrophysical Journal*, 882(1):56, 2019. doi: 10.3847/1538-4357/ab3124.

R. Rieger and K. Müllen. Forever young: Polycyclic aromatic hydrocarbons as model cases for structural and optical studies. *Journal of Physical Organic Chemistry*, 23(4):315–325, 2010. doi: 10.1002/poc.1644.

Fred J. Rispoli. Counting perfect matchings in hexagonal systems associated with benzenoids. *Mathematics Magazine*, 14:194–200, 2001.

F. Rossi, P. van Beek, and T. Walsh. *Handbook of Constraint Programming*. Elsevier, 2006.

Myriam Roy, Veronika Berezhnaia, Marco Villa, Nicolas Vanthuyne, Michel Giorgi, Jean-Valère Naubron, Salomé Poyer, Valérie Monnier, Laurence Charles, Yannick Carissan, Denis Hagebaum-Reignier, Jean Rodriguez, Marc Gingras, and Yoann Coquerel. Stereoselective Syntheses, Structures, and Properties of Extremely Distorted Chiral Nanographenes Embedding Hextuple Helicenes. *Angewandte Chemie International Edition*, 59(8):3264–3271, 2020.

Yosadara Ruiz-Morales. The Agreement between Clar Structures and Nucleus-Independent Chemical Shift Values in Pericondensed Benzenoid Polycyclic Aromatic Hydrocarbons: An Application of the Y-Rule. *J. Phys. Chem. A*, 108:10873–10896, 2004.

J.-C. Régin. A filtering algorithm for constraints of difference in CSPs. In *Proceedings of AAAI*, pages 362–367, 1994.

Ana Sánchez-Grande, José I. Urgel, Libor Veis, Shayan Edalatmanesh, José Santos, Koen Lauwaet, Pingo Mutombo, José M. Gallego, Jiri Brabec, Pavel Beran, Dana Nachtigallová, Rodolfo Miranda, Nazario Martín, Pavel Jelínek, and David Écija. Unravelling the Open-Shell Character of Peripentacene on Au(111). *The Journal of Physical Chemistry Letters*, 12(1):330–336, 2021. doi: 10.1021/acs.jpclett.0c02518.

Paloma Vieira Silva and Eduardo Costa Girão. Electronic and Transport Properties of Graphene Nanoribbons Based on Super-Heptazethrene Molecular Blocks. *The Journal of Physical Chemistry C*, 125(20):11235–11248, 2021. doi: 10.1021/acs.jpcc.1c02514.

David Simoncini, David Allouche, Simon de Givry, Céline Delmas, Sophie Barbe, and Thomas Schiex. Guaranteed Discrete Energy Optimization on Large Protein Design Problems. *Journal of Chemical Theory and Computation*, 11 (12):5980–5989, 2015. doi: 10.1021/acs.jctc.5b00594.

Peter R. Taylor. Molecular symmetry and quantum chemistry. In Björn O. Roos, editor, *Lecture Notes in Quantum Chemistry: European Summer School in Quantum Chemistry*, Lecture Notes in Chemistry. 1992.

Robert J. Ternansky, Douglas W. Balogh, and Leo A. Paquette. Dodecahedrane. *Journal of the American Chemical Society*, 104(16):4503–4504, 1982. doi: 10.1021/ja00380a040.

Georges Trinquier and Jean-Paul Malrieu. Predicting the Open-Shell Character of Polycyclic Hydrocarbons in Terms of Clar Sextets. *The Journal of Physical Chemistry A*, 122(4):1088–1103, 2018. doi: 10.1021/acs.jpca.7b11095.

Mizuho Uryu, Taito Hiraga, Yoshito Koga, Yutaro Saito, Kei Murakami, and Kenichiro Itami. Synthesis of Polybenzoacenes: Annulative Dimerization of Phenylene Triflate by Twofold C-H Activation. *Angew. Chem.*, 132(16):6613–6616, 2020. ISSN 1521-3757. doi: 10.1002/ange.202001211.

Christine Wei Wu. Modelling Chemical Reactions Using Constraint Programming and Molecular Graphs. In *Principles and Practice of Constraint Programming*, pages 808–808, 2004.

Jishan Wu, Wojciech Pisula, and Klaus Müllen. Graphenes as Potential Material for Electronics. *Chemical Reviews*, 107 (3):718–747, 2007. doi: 10.1021/cr068010r.

Zeming Xia, Sai Ho Pun, Han Chen, and Qian Miao. Synthesis of Zigzag Carbon Nanobelts through Scholl Reactions. *Angew. Chem. Int. Ed.*, 60(18):10311–10318, 2021. ISSN 1521-3773. doi: 10.1002/anie.202100343.

A Detailed Results for The Subset $\mathcal{B}_1$

Figures 28-33 describe the benzenoid structures considered in the subset $\mathcal{B}_1$. These structures have various sizes or shapes. They may admit or not some symmetries. Moreover, in these figures, we also specify the values computed by CRECP and NICS in blue and red respectively. For sake of readability, we only provide them for a single hexagon per symmetry class. Indeed, all the hexagons of a symmetry class have the same value whatever the considered method.

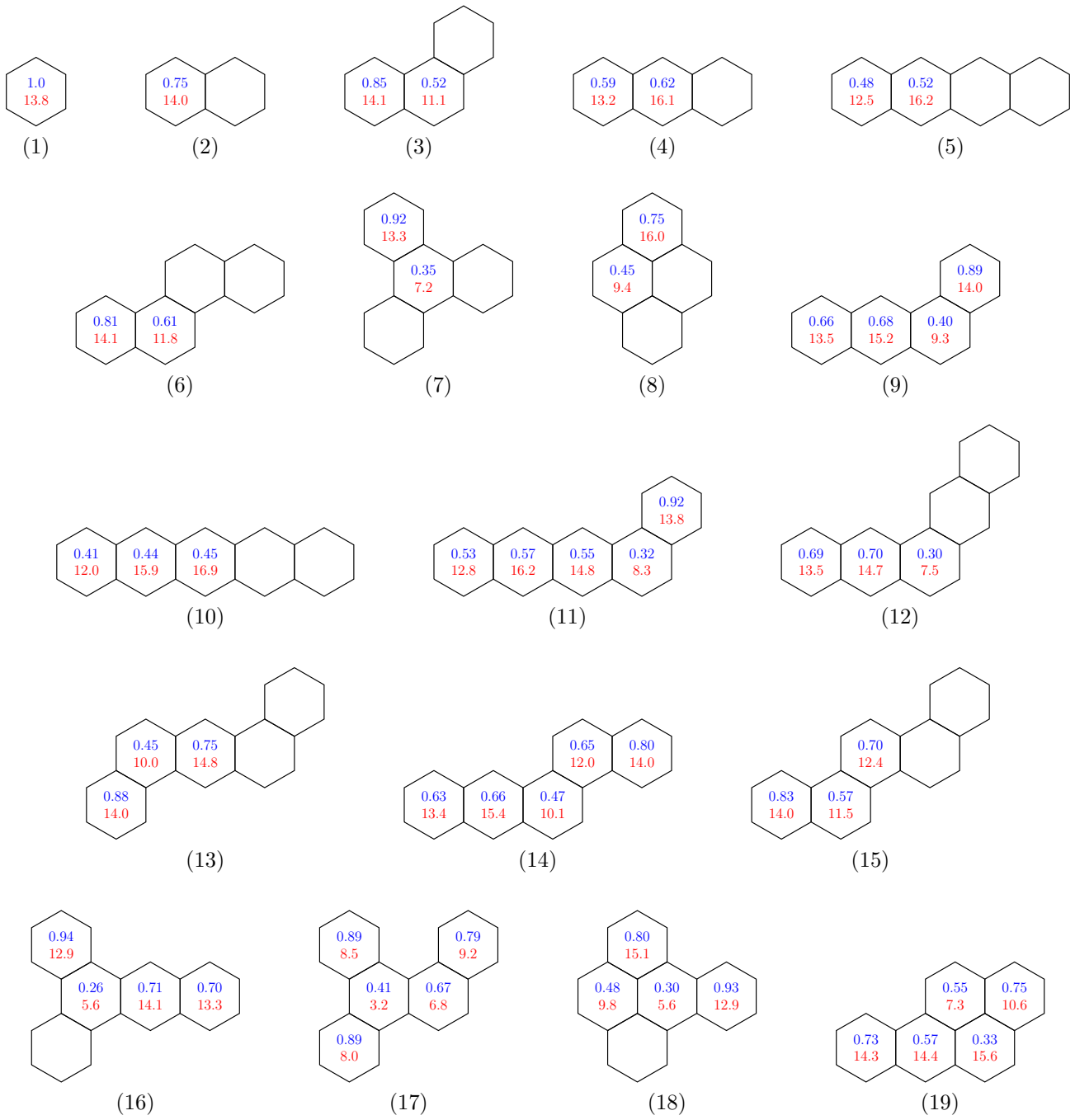
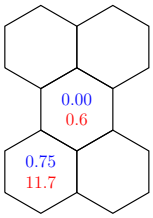
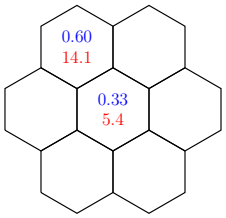


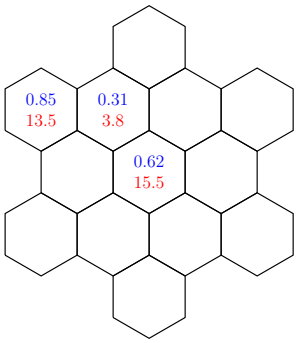
Figure 28: Results on the set of 48 molecules of the subset $\mathcal{B}_1$.



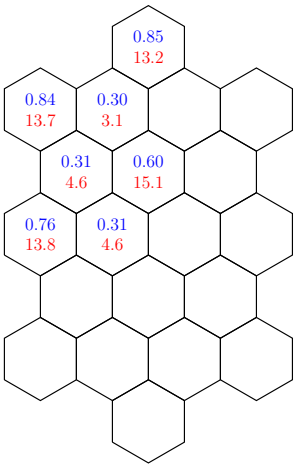
(20)



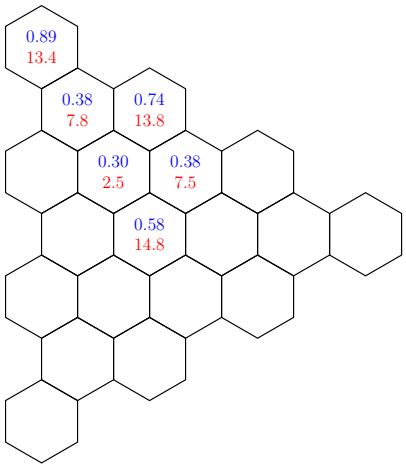
(21)



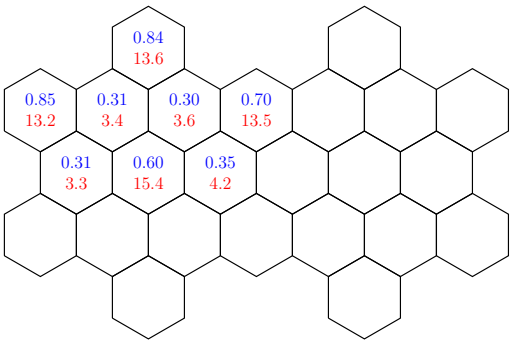
(22)



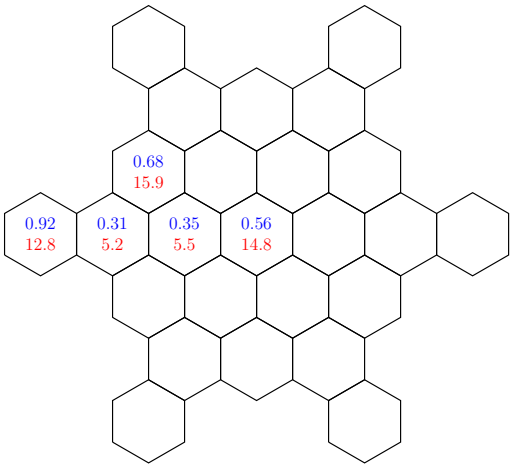
(23)



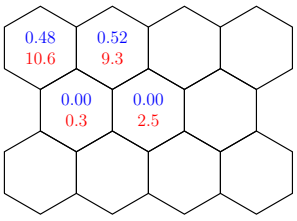
(24)



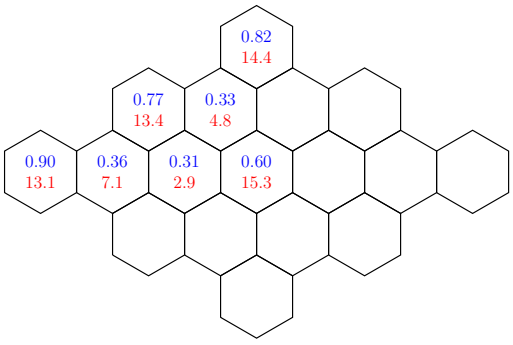
(25)



(26)



(27)



(28)

Figure 29: Results on the set of 48 molecules of the subset $\mathcal{B}_1$ (Figure 28 continued).

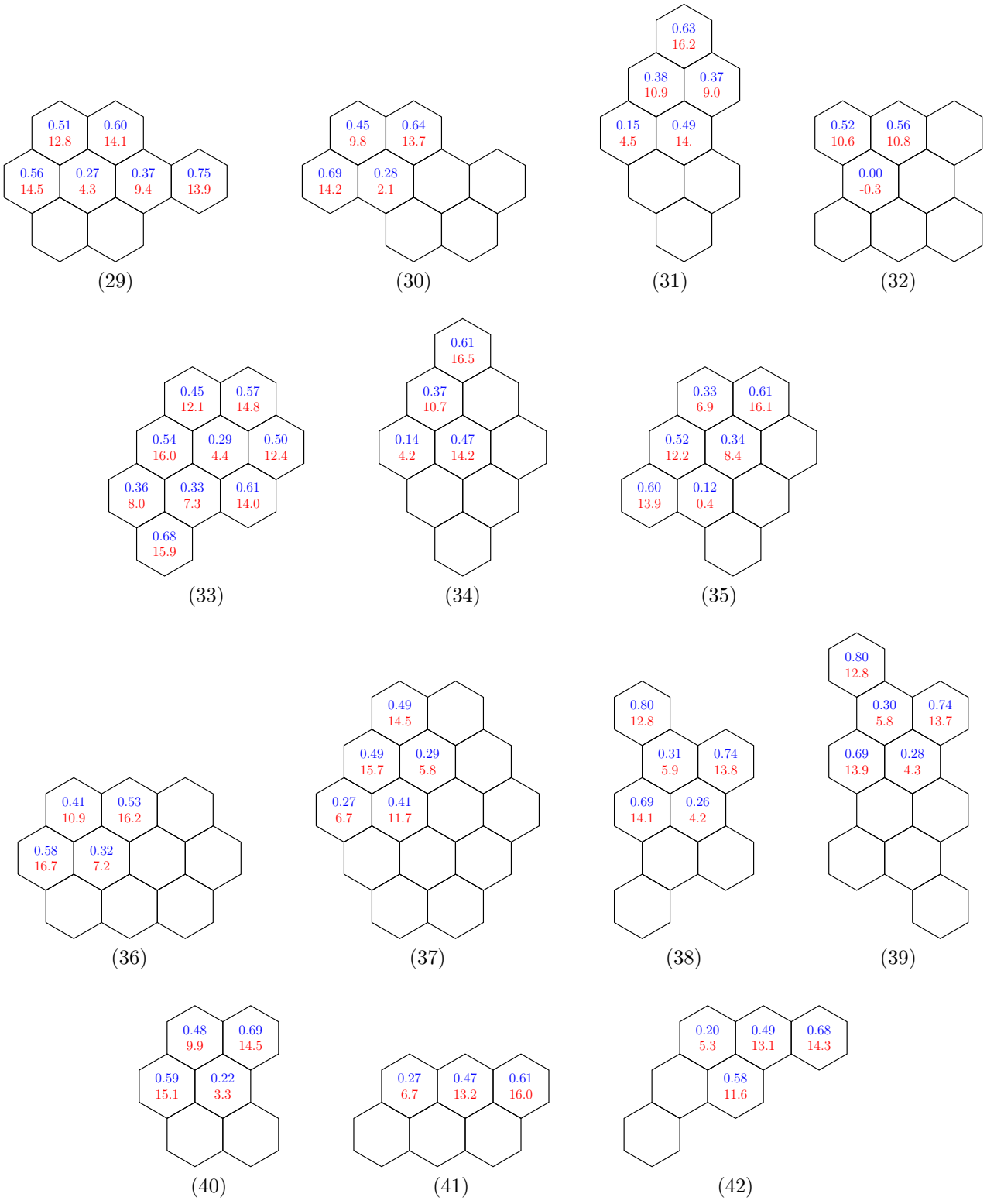
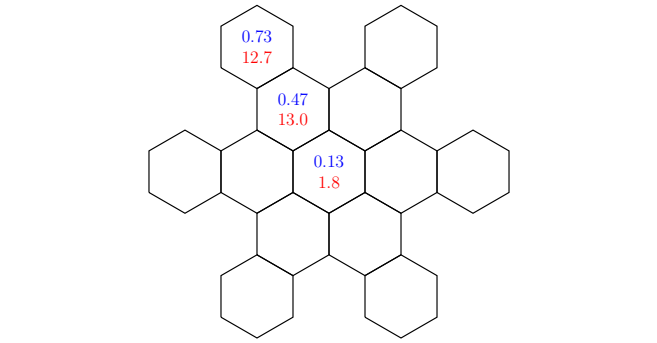
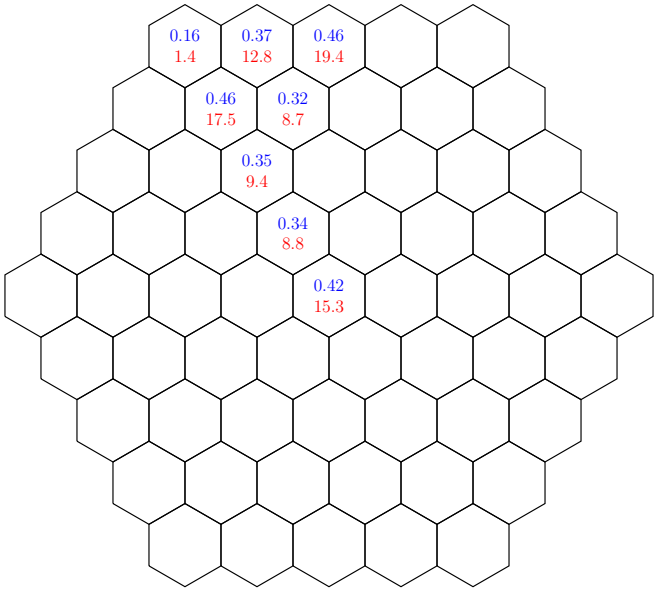


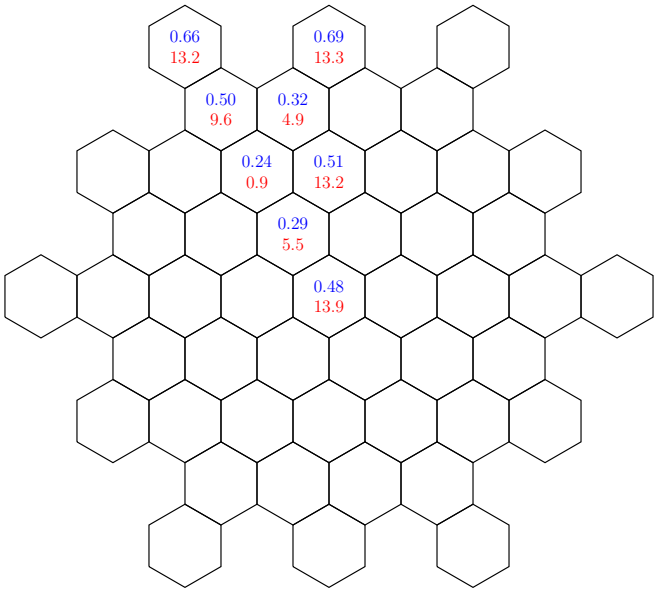
Figure 30: Results on the set of 48 molecules of the subset $\mathcal{B}_1$ (Figure 29 continued).



(43)

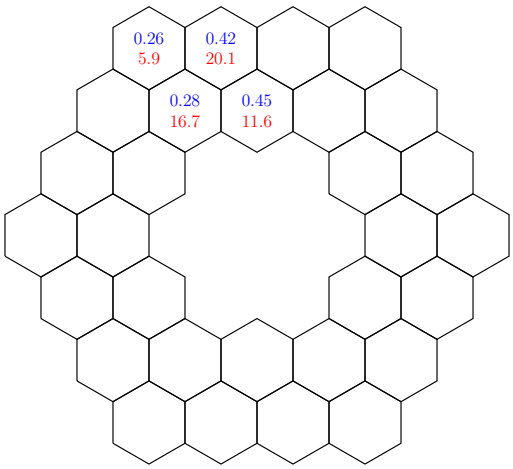


(44)

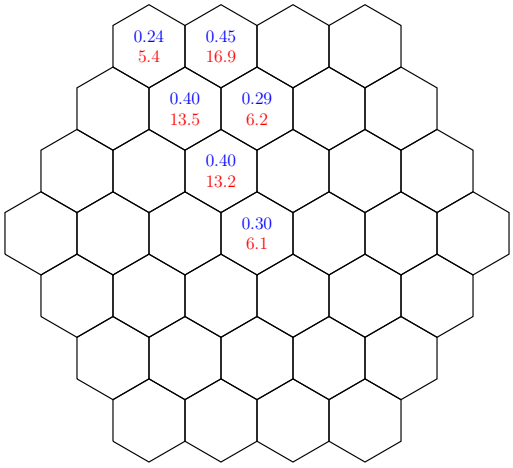


(45)

Figure 31: Results on the set of 48 molecules of the subset $\mathcal{B}_1$ (Figure 30 continued).

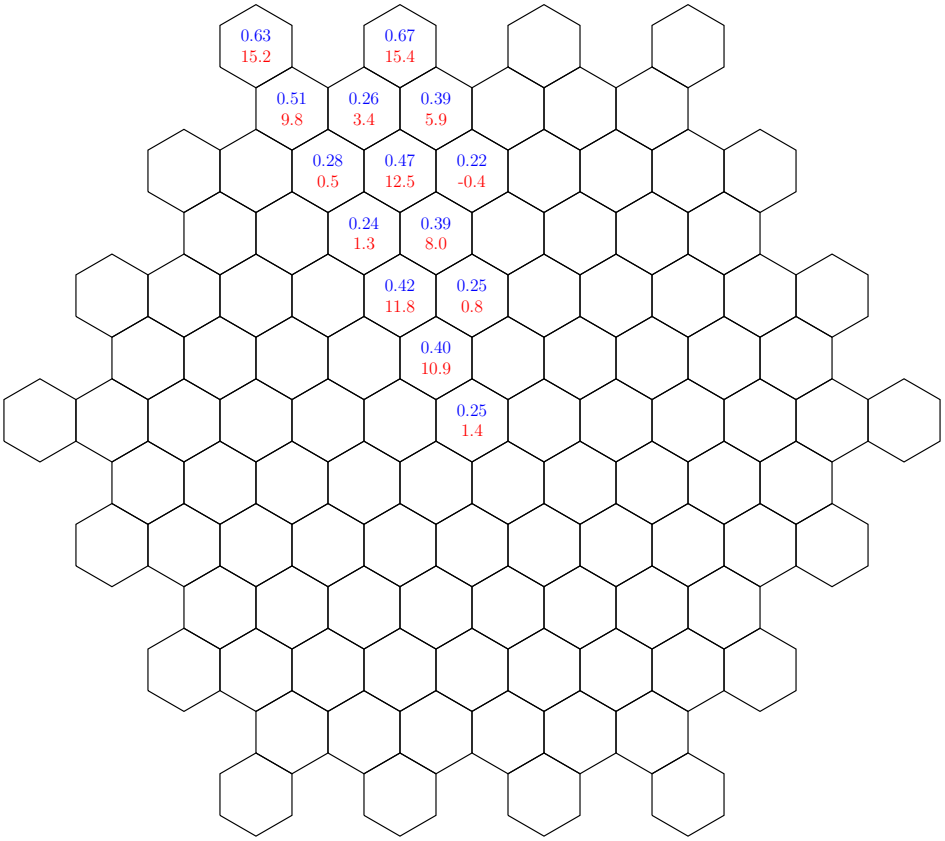


(46)



(47)

Figure 32: Results on the set of 48 molecules of the subset $\mathcal{B}_1$ (Figure 31 continued).



(48)

Figure 33: Results on the set of 48 molecules of the subset $\mathcal{B}_1$ (Figure 32 continued).