# Using Constraint Programming to Generate Benzenoid Structures in Theoretical Chemistry

Yannick Carissan, Denis Hagebaum-Reignier, Nicolas Prcovic, Cyril Terrioux, Adrien Varet

# Using Constraint Programming to Generate Benzenoid Structures in Theoretical Chemistry*,**

Yannick Carissan[1][0000−0002−9876−0272], Denis Hagebaum-Reignier[1][0000−0001−8761−1047], Nicolas Prcovic[2], Cyril Terrioux[2][0000−0002−9779−9108], and 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. Therefore, generating benzenoids which have certain structural properties (e.g. having a given number of hexagons or having 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.

**Keywords:** Constraint programming · modeling · Graph variables and constraints · Chemistry

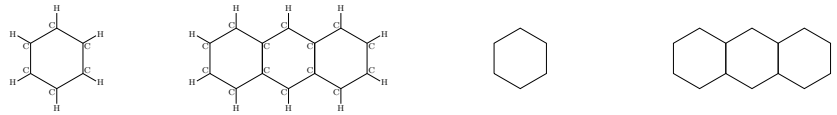
## 1 Introduction

*Polycyclic aromatic hydrocarbons (PAHs)* are hydrocarbons whose carbons 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 natural environment, these molecules

* This work has been funded by the Agence Nationale de la Recherche project ANR-16-C40-0028.

** The final authenticated version is available online at https://doi.org/10.1007/978-3-030-58475-7_40.



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

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

are created by the incomplete combustion of carbon contained in combustibles [18]. They are popular research subjects in material sciences, e.g. molecular nanoelectronics where they are used to store or transport energy [29,2] or in organic synthesis [25,22], 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 [14].

PAHs 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 size, 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 explore the effect of the size, shapes in terms of compacity and symmetry of PAHs, on band positions and intensities of the infrared spectra [1,3,23]. Very recently, a systematic investigation of a series of 328 PAHs containing up to 150 carbon atoms showed that PAHs with armchair edges 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 [24]. 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 [8]. Constraint-programming is particularly well suited for the generation of such families of PAHs.

Another important example where the generation of specific shapes is relevant for chemists deals with so-called "non-Kekulean" benzenoids [10]. 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. It was accepted among chemists until recently that "non-Kekulean" benzenoids should be 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 their synthesis would be a real challenge. The experimental realization and in-depth characterization of small "non-Kekulean" benzenoids was very recently achieved on gold surfaces [21,20]. These studies opened the way to the synthesis of new classes of compounds which 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 [27]. From a quantum theoretical 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 having or not a particular shape or satisfying a particular property (e.g. [5,6]). These are bespoke approaches which have the advantage of being efficient, but 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 efforts. So, in this paper, we prefer to use an 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 so doing, our approach benefits from the flexibility of CP and requires less efforts of implementation. In the meantime, CP offers efficient solvers which can be quite competitive with respect to bespoke algorithms.

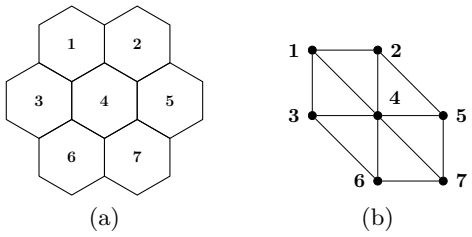
The paper is organized as follows. First, we recall some definitions about benzenoids and constraint programming in Section 2. Section 3 introduces the fastest existing algorithm for generating benzenoids. Then Section 4 presents a new approach using constraint programming, explains its advantages and gives some examples. Finally, we conclude and provide some perspectives in Section 5.

## 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 benzenic rings. For example, Figure 1(b) shows anthracene, which contains three benzenic rings.

By definition of the *valence* (i.e. the number of bonds that an atom can establish) of carbon and hydrogen atom, we know that each carbon atom is




(a) (b)

**Fig. 2.** Coronene (a) and its hexagon graph (b).

linked to either two other carbon atoms and one hydrogen atom or three other carbon atoms. So 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 a consequence, 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 the following, 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. Figure 2(a) presents the graph corresponding to coronene (a well-known benzenoid). This graph has eight faces namely the seven numbered faces and the external face. Note that in the sequel, we do not take into account the external face. For this example, the numbered faces correspond exactly to the hexagons of coronene. However, we will see later that this property does not hold for all the benzenoids.

Then, 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(b) presents the hexagon graph of coronene. The hexagon graph allows us to express the interaction between the hexagons of the considered benzenoid.

### 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. $C = \{c_1, ..., c_e\}$ represents a set of $e$ *constraints*. Constraints represent the interactions between the variables and describe the allowed combinations of values.

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

Many libraries are available to represent and solve efficiently CSP instances. In this paper, we exploit the open-source Java library *Choco* [15]. 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 Benzenoids

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. Of course, the more interesting instances of the BGP problem combine several properties. For example, Figure 5 shows benzenoids having a tree as hexagon graph. Such a property-based instances design 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 [27]), which arise from their geometrical arrangement.

Now, we present an existing method proposed by Brinkmann et al. [5]. 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 $n$ vertices. This is done by adding successively new vertices to the hexagon graph (which is equivalent to generate all the wanted molecules by successively adding new hexagons).

This method is really efficient. For instance, it could generate the 669,584 benzenoids having 12 hexagons in 1.2 seconds and 1,000 billions 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 which does not correspond to a hexagon or the external face. For example, Figure 3(a) depicts the smallest benzenoid (in terms of 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 to close a cycle of hexagons. If so, the hexagon is not added and so benzenoids with holes cannot be generated. Benzenoids with holes are quite seldom. 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 Table 1). Furthermore, this method is unable to take into account other properties natively and cannot easily be tuned

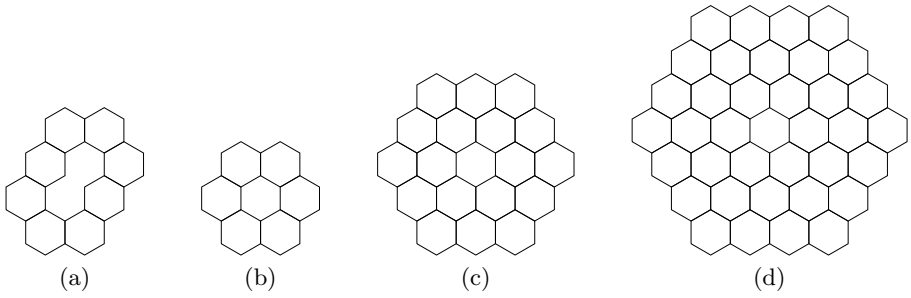



**Fig. 3.** The smallest benzenoid with hole (a) and coronenoids of size 2 (b), 3 (c) and 4 (d).

to fit the needs of chemists. Indeed, it is based on an augmenting procedure that decides how to add a vertex. So this procedure should be changed and proven adequate to avoid generating non canonical graphs (i.e. redundant isomorphic graphs) each time we want to change the structural property of the benzenoids we wish to generate. It is therefore 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 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 molecule of benzene (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 3(b)-(d) present the coronenoids of size 2, 3 and 4. 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 for 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 $k(n) = \left\lfloor \frac{n+1}{2} + 1 \right\rfloor$.

So if we reason in terms of hexagon graph, obtaining all the benzenoids with $n$ hexagons is equivalent to find all the connected sub-graphs of the hexagon graph of coronenoid of size $k(n)$. The model we propose relies on this property.

So, 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 coronenoid of size $k(n)$ (see Figure 4(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 coronenoid of size $k(n)$. The variable $x_i$ is set to 1 if the $i$th hexagon of coronenoid of size $k(n)$ is used in the hexagon graph depicted by $x_G$, 0 otherwise. For sake of simplicity, hexagons are numbered from top to bottom and from left to right like in Figure 2. Likewise, we consider a set of $m_c$ Boolean variables $\{y_1, \ldots, y_{m_c}\}$ where $m_c$ is the number of edges of the hexagon graph of coronenoid of size $k(n)$. The variable $y_j$ is set to 1 if the $j$th edge of the hexagon graph of coronenoid of size $k(n)$ is used in the hexagon graph depicted by $x_G$, 0 otherwise. We must emphasize that the set of $x_i$ and $y_i$ variables and the channeling constraints maintaining the consistency between their values and the value of $x_G$ are automatically generated by Choco Solver through the call of a predefined method.

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 $i$th hexagon of coronenoid of size $k(n)$ is used in the graph represented by $x_G$. So we must ensure that their respective values are consistent each others. For 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 $y_j$* Like previously, we consider a channeling constraint per variable $y_j$ which involves $y_j$ and $x_G$ and imposes that $y_j = 1 \iff x_G$ *contains the edge* $j$.
- *$x_G$ 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(b), removing only edge $\{1, 2\}$ does not produce a valid hexagon graph. To ensure that the hexagon graph is valid, we must add a constraint for every triplet $(h_{j_1}, h_{j_2}, h_{j_3})$ of hexagons which are pairwise adjacent in the coronenoid hexagon graph. This constraint imposes that if two of these edges exists, then the third one exists too. This can be achieved by posting a set of ternary clauses of the form $\{\neg y_{j_1} \vee \neg y_{j_2} \vee y_{j_3}, y_{j_1} \vee \neg y_{j_2} \vee \neg y_{j_3}, \neg y_{j_1} \vee y_{j_2} \vee \neg y_{j_3}\}$ for each possible triple of pairwise adjacent hexagons.
- *Benzenoids have $n$ hexagons* It can be easily done 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 $x_G$ ensures 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 forms by the hexagons 1, 2, 5, 7, 6 and 3 of coronene (see Figure 2), we have necessarily a hexagon in the middle of the crown, namely the hexagon 4. To ensure this property, we add a set of constraints which specify 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: $\sum_{v \in N(u)} x_v = 6 \Rightarrow x_u = 1$.

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 which 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. So, we have to create a constraint which specifies that the sum of the binary variables associated to the top border (resp. left border) is strictly positive.
– A set of constraints which 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 [11]. For each of the twelve symmetries, it requires $n_c$ new Boolean variables (each associated with a $x_i$ Boolean variable representing a hexagon) and a total of $3n_c$ ternary clauses.

This model can be easily implemented with the open-source Java library *Choco* [15]. Indeed, Choco natively proposes graph variables and the more usual graph-related constraints (notably `connected` constraint).

### 4.2 How to Specialize the Model

The first advantages 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 it easier the addition of most of structural properties wished by the chemists. Indeed, starting from the general model, for each new desired property, we simply have to model it by posting new constraints and eventually by adding new variables.

For example, let us consider that chemists are interested by benzenoids whose structure is a path of hexagons. Such benzenoid structures can easily generated by exploiting the general model $I$ and adding the graph constraint `path` on $x_G$. Now, if chemists are more interested by *catacondensed benzenoids*, that is benzenoids whose structure is a tree, we can just add the graph constraint `tree`

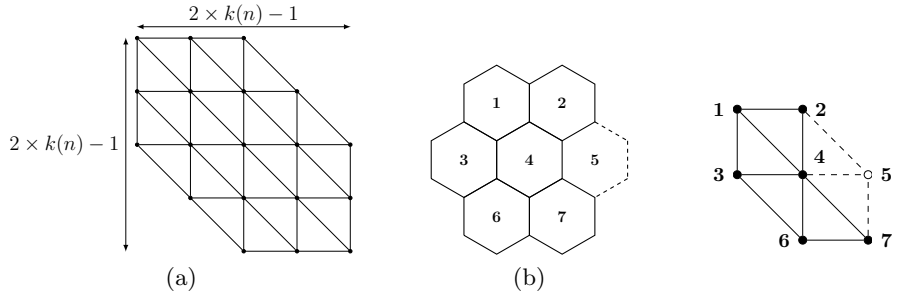



**Fig. 4.** Upper bound of the domain of the graph variable (a), rectangle benzenoid (in solid line) of dimension $3 \times 2$ embedded in coronenoid of size 2 (b) and its related hexagon graph (c).

on $x_G$ to the general model $I$. Figure 5 shows nine (among twelve possible) examples of 5-hexagon benzenoids obtained by just adding the `tree` constraint of Choco on $x_G$.

Of course, depending on the desired property the model may be more complex. Especially, it may require to add new variables or the property cannot be directly expressed by a single existing constraint. In next subsections, we give such examples.

### 4.3 Generating Rectangle Benzenoids

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"*, and add it to the model we describe previously. For instance, Figure 6(i) shows a rectangle benzenoid with the dimensions $3 \times 3$.

First, remember that the general model described in the previous part takes in 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 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 coronenoid of size $k(n)$). Figure 4 shows the rectangle benzenoid of dimensions $2 \times 3$ embedded in coronenoid of size 2 (b) and its hexagon graph (c).

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 $\{1, ..., w_{max}\}$ and $\{1, ..., h_{max}\}$. 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 $x_i$ which appear in the $i$th line (resp. $i$th column) in the coronenoid of order $k(n)$. We assume that lines (resp. columns) are numbered from top to bottom (resp. from left to right). For example, if we consider the hexagon graph of the rectangle benzenoid of dimensions $3 \times 2$

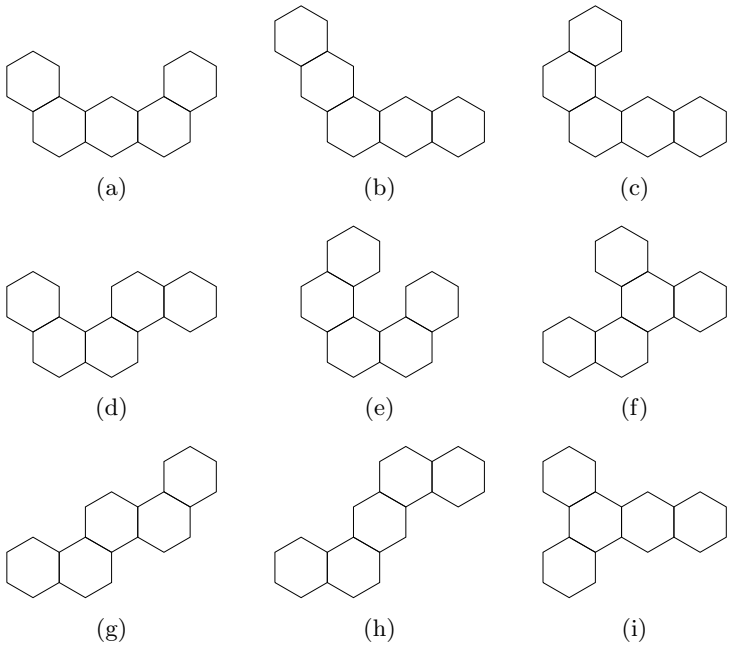



**Fig. 5.** Catacondensed benzenoids with 5 hexagons.

described in Figure 4(b), we have the following sets:

$$\begin{cases} L_1 = \{x_1, x_2\} \\ L_2 = \{x_3, x_4, x_5\} \\ L_3 = \{x_6, x_7\} \\ C_1 = \{x_1, x_3, x_6\} \\ C_2 = \{x_2, x_4, x_7\} \end{cases}$$

Now 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 to have a Boolean variable equal to 0 between two Boolean variables equal to 1. For the $i$th line, this can be modeled by imposing an arithmetic constraint $x_{i_1} \geq x_{i_2} \ldots \geq x_{i_{wmax}}$ if $L_i = \{x_{i_1}, x_{i_2}, \ldots, x_{i_{wmax}}\}$. We can also use instead a global constraint `ordered` applied on the variables of $L_i$ with operator $\geq$.
- *The hexagons of each column are positioned contiguously* We proceed as for the lines by considering $C_i$ 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 $i$th line, we add a constraint linking $x_w$ to all the variables in $L_i$ and imposing $\sum_{x_{i_j} \in L_i} x_{i_j} = 0 \vee \sum_{x_{i_j} \in L_i} x_{i_j} = x_w$. As such a

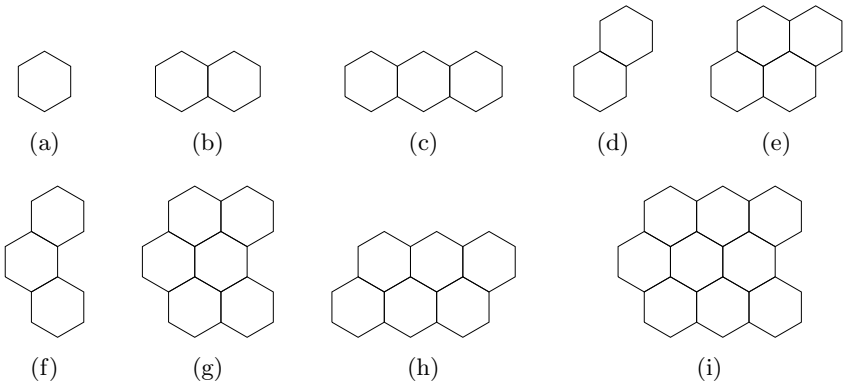



**Fig. 6.** Rectangle benzenoids generated with $w = h = 3$.

constraint is added for each line, we are sure that all the lines have the same width.

– *Columns have a consistent size* We proceed as for the lines by considering $C_i$ instead of $L_i$ and $x_h$ instead of $x_w$.

Figure 6 shows the 9 rectangle benzenoids generated with the parameters $w = h = 3$.

In this extended model, we can note that some variables are now useless. Indeed, only the leftmost hexagons of coronenoid of order $k(n)$ covered by the rectangle of dimensions $w_{max} \times h_{max}$ are required. The other hexagons will only lead to produce symmetrical structures. So, we can refine our model by removing useless variables. Likewise, we can filter the domain of $x_G$ in order that $GUB$ is the hexagon graph of the rectangle $w_{max} \times h_{max}$ by posting the adequate unary constraint.

This extension of our general model is given as a simple illustration of our approach. Of course, we can easily generate benzenoids having a rectangle shape with a bespoke algorithm. What is interesting in our approach is its flexibility. For instance, if some chemists are interested by identifying the rectangular benzenoids which has a given Clar number, we have only 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 [9]. Unfortunately, due to page limit, we cannot detail the corresponding extended model.

### 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 [12,4,13]. Their enumeration and generation gave rise to several studies (e.g., [6] which enumerates 2-hole coronoids and

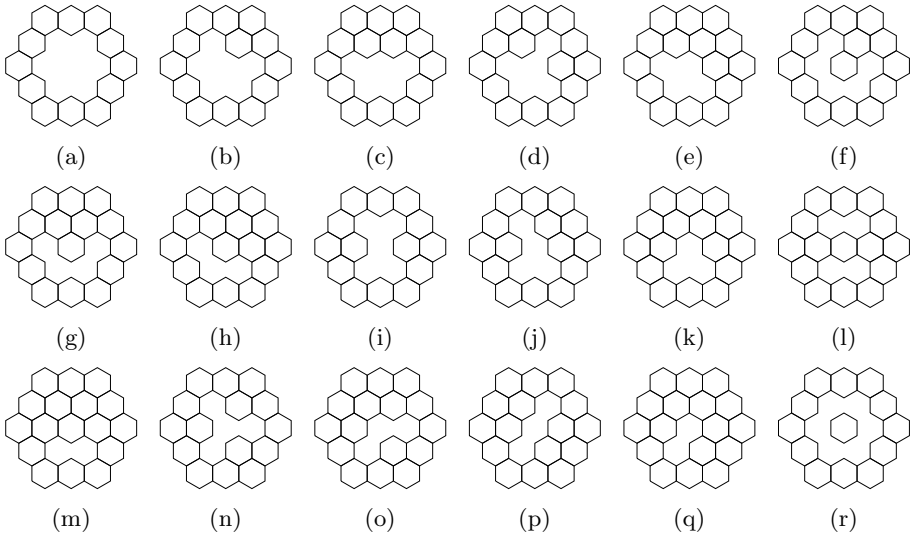



**Fig. 7.** All the ways of digging holes in the coronenoid of size 3. The last one (r) is not a valid coronoid: the hole in the coronenoid is cyclic and has disconnected the "coronoid" into two benzenoids. Hence, the constraint that $x_C$ must be connected.

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 try all possible ways to add hexagons around. The second kind of approach consists of generating all the benzenoids with $n$ hexagons and then detects the ones with holes. Another possible approach consists in generating benzenoids without holes (e.g. with the method of Brinkmann et al. [5]) 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 which would directly generate coronoids. Indeed, if the number of hexagons is increased by one, the number of benzenoids is multiplied by approximately 5 (as well as the time to generate them [5]), whereas the time to generate the coronoids with the direct approach we describe below appears to be only twice longer (see Table 1). So, in this part, we present how we can model the property *"all the built benzenoids have a hole and are contained in a benzenoid with n hexagons"*. This allows to generate easily all kinds of coronoids with any number of holes.

Any coronoid can be seen as a benzenoid that has lost several contiguous hexagons (which created holes). So, the vertices of the benzenoid can be split into the vertices belonging to a coronoid and the vertices forming holes. To model this problem, we consider our general model. First, we define two new graph variables $x_C$, which represents an underlying coronoid of $x_G$, and $x_H$ the holes to dig in $x_G$ to form $x_C$. $x_C$ and $x_H$ have the same domain as $x_G$. So, we can have all the possible coronoid $x_C$ by generating all the pairs $(x_G, x_H)$. There

**Table 1.** Number of coronoids obtained by digging holes from all the benzenoids with $n$ hexagons.

| $n$ | #coronoids | time (s) | #benzenoids without holes |
|---|---|---|---|
| 10 | 1 | 43 | 30,086 |
| 11 | 4 | 114 | 141,229 |
| 12 | 38 | 262 | 669,584 |
| 13 | 239 | 533 | 3,198,256 |
| 14 | 1,510 | 1,076 | 15,367,577 |

can be several values of $x_H$ for a value of $x_G$, as illustrated in Figure 7. Then we consider two sets of $n_c$ Boolean variables $\{x_1^C, \ldots, x_{n_c}^C\}$ and $\{x_1^H, \ldots, x_{n_c}^H\}$ (with $n_c$ the number of hexagons of coronenoid of size $k(n)$). Like $x_i$ for $x_G$, the variable $x_i^C$ (resp. $x_i^H$) is set to 1 if the $i$th hexagon of coronenoid of size $k(n)$ is used in the graph depicted by $x_C$ (resp. $x_H$), 0 otherwise. Likewise, we define the set of $m_c$ Boolean variables $\{y_1^H, \ldots, y_{m_c}^H\}$ (with $m_c$ the number of edges in the hexagon graph of coronenoid of size $k(n)$). The variable $y_j^H$ is set to 1 if the $j$th edge of coronenoid of size $k(n)$ is used in the graph depicted by $x_H$, 0 otherwise.

Finally, we add the following constraints ensuring that variables $x_H$ and $x_C$ have the right properties:

- $x_H$ *is a sub-graph of* $x_G$ This is enforced thanks to the `subgraph` constraint of Choco applied on variables $x_H$ and $x_G$.
- *Only fully surrounded vertices of* $x_G$ *can be vertices of* $x_H$ For all vertices of $x_H$ if the degree of a vertex in $x_H$ is strictly positive then the degree of the same vertex in $x_G$ is 6. Indeed, only the vertices/hexagons in $x_G$ surrounded by six hexagons can belong to a hole. This constraint is enforced thanks to clauses on the $y_i$ and $y_i^H$ Boolean variables.
- *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 $x_H$.
- $x_H$ *involves at least two hexagons.* We post the constraint $\sum_{i \in \{1,\ldots,n_c\}} x_i^H > 1$.
- $x_C$ *and* $x_H$ *form a partition of* $x_G$ *w.r.t. hexagons* For all vertices of $x_G$, a vertex is in $x_C$ iff this vertex is in $x_G$ and not in $x_H$. This ensures that any vertex of $x_G$ is either in $x_C$ or in $x_H$. With this aim in view, we add a clause $x_i^C \leftrightarrow (x_i \wedge \overline{x_i^H})$ on $x_i$, $x_i^C$ and $x_i^H$ for any $i \in \{1, \ldots, n_c\}$.
- $x_C$ *is connected* If $x_C$ is not connected, we may obtain two benzenoids instead of one (see Figure 7(r) for instance). Again, this can be achieved by exploiting the graph constraint `connected` applied on $x_C$.

For example, Figure 8(a) shows how the coronoid of Figure 9(a) can be embedded in the coronenoid of size 3. Then, we depict in dashed line the value of $x_G$, $x_C$ and $x_H$ respectively in Figures 8(b)-(d).

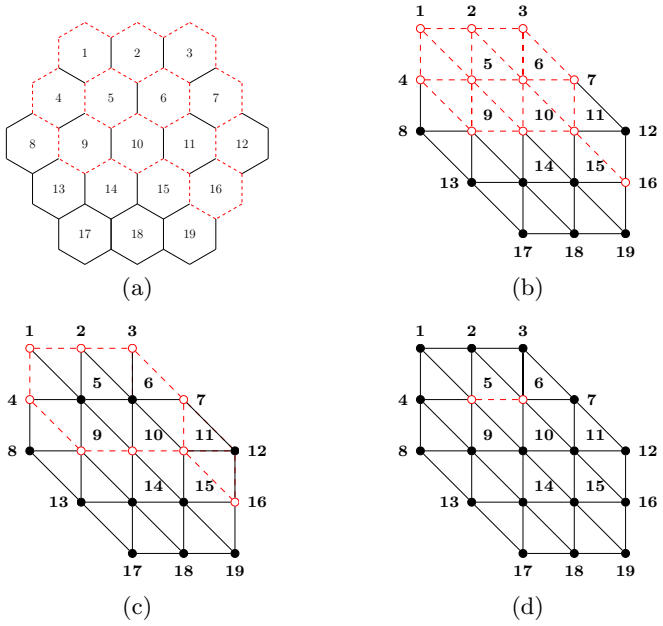



**Fig. 8.** The coronoid of Figure 9(a) embedded in the coronenoid of size 3 (a),the corresponding value of $x_G$ (b), $x_C$ (c) and $x_H$ (d).

Table 1 shows the results of the experiments we ran on a 3.4 GHz Intel Core i7 iMac with a 12 Gb RAM. We implemented our CSP model in Java 8 with Choco Solver 4.0.4 using the choco-graph 4.2.3 module. We did not specify any search strategy or heuristic, so the default ones were used by the search engine. We generated the coronoids by digging holes in different sizes of benzenoids. Among all the benzenoids with $n$ hexagons, we show the number of coronoids we can produce by removing hexagons. For example, the only coronoid produced from the 10-hexagon benzenoids is the 8-hexagon coronoid of Figure 3(a). Figure 9 lists the four coronoids for $n = 11$. To show how rare coronoids are, Table 1 also provides the number of benzenoids without holes [5]. Of course, thanks to the model we propose for coronoid generation, we do not consider all these benzenoids. Indeed, they are not generated by Choco Solver because it filters out benzenoids that cannot have holes through constraint propagation.

### 4.5 Generating Symmetric Benzenoids

Benzenoids are also classified by chemists by their classes of internal symmetries (symmetries that let a benzenoid invariant by rotation and/or mirroring). We can generate such classes of benzenoids by adding the constraints for enforcing internal symmetries. When searching for all the possible benzenoids embeddable in a coronenoid of size 3, we obtain the 11,578 benzenoids (with at most 19

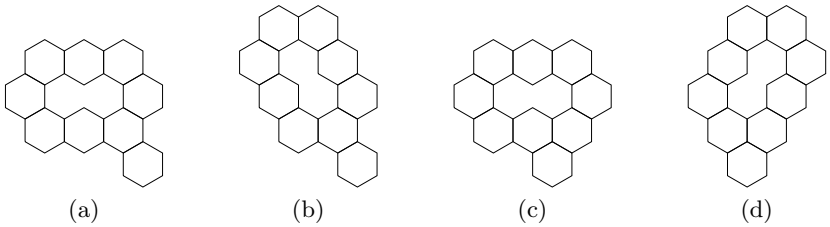



**Fig. 9.** Coronoids for $n = 11$.

hexagons) in 36 minutes. Enforcing invariance by 60 degree rotation (to obtain the 4 corresponding benzenoids), by 120 degree rotation (16 benzenoids) and 180 degree rotation (70 benzenoids) takes less than one second for each task. This strengthens the idea that constraint propagation in nowadays solvers is efficient enough to allow these theoretical chemistry problems to be modeled and solved with CP without having to define and use bespoke methods. Moreover, interestingly, note that this extension of our general model may be combined with any extension described above.

## 5 Conclusions and Perspectives

In this paper, we addressed the problem of generating benzenoid structures, which is an interesting and important problem in theoretical chemistry. In this context, we presented an approach using constraint programming able to generate benzenoids which satisfy a certain amount of properties. Its main advantage w.r.t. existing methods in the literature lies in its flexibility. Indeed, from a general model, we can express additional properties by simply adding variables and/or constraints while existing bespoke methods rely on more rigid and complex notions and cannot be adapted without requiring heavy tasks. Moreover, our approach turns to be more general, making it possible to generate benzenoids with holes for instance.

Chemists are interested in generating benzenoids with particular shapes (e.g. rectangle or rhombus shapes [27]). We have already dealt with the rectangle shapes in this paper. So a natural extension of this work relies in taking into account other specific properties related to the needs of chemists. Another step consists in studying the limit of our approach both in terms of properties we can express and our ability to generate benzenoids of large size. Furthermore, this paper shows how, once again, constraint programming can be useful to tackle and solve problems related to theoretical chemistry [19,28,26,16,17]. In particular, many questions about benzenoids can be modeled as decision or optimization problems under constraints (e.g. computing their aromaticity or finding the closest structure to a Kekulé structure) and can correspond to difficult tasks (e.g. computing the Clar number is NP-hard [7]). It could be of interest for both communities to study them.

## References


1. Allamandola, L.J., Hudgins, D.M., Sandford, S.A.: Modeling the Unidentified Infrared Emission with Combinations of Polycyclic Aromatic Hydrocarbons. The Astrophysical Journal **511**(2), L115–L119 (1999). https://doi.org/10.1086/311843
2. Aumaitre, C., Morin, J.F.: Polycyclic Aromatic Hydrocarbons as Potential Building Blocks for Organic Solar Cells. The Chemical Record **19**(6), 1142–1154 (2019). https://doi.org/10.1002/tcr.201900016, https://onlinelibrary.wiley.com/doi/abs/10.1002/tcr.201900016
3. Bauschlicher, Jr., C.W., Peeters, E., Allamandola, L.J.: The Infrared Spectra of Very Large, Compact, Highly Symmetric, Polycyclic Aromatic Hydrocarbons (PAHs). The Astrophysical Journal **678**(1), 316–327 (2008). https://doi.org/10.1086/533424
4. Beser, U., Kastler, M., Maghsoumi, A., Wagner, M., Castiglioni, C., Tommasini, M., Narita, A., Feng, X., Müllen, K.: A C216-Nanographene Molecule with Defined Cavity as Extended Coronoid. Journal of the American Chemical Society **138**(13), 4322–4325 (2016). https://doi.org/10.1021/jacs.6b01181
5. Brinkmann, G., Caporossi, G., Hansen, P.: A Constructive Enumeration of Fusenes and Benzenoids. Journal of Algorithms **45(2)** (2002)
6. Brunvoll, J., Cyvin, R.N., Cyvin, S.J.: Enumeration and Classification of Double Coronoid Hydrocarbons – Appendix: Triple Coronoids. Croatica Chemica Acta **63**(4), 585–601 (1990)
7. Bérczi-Kovács, E., Bernáth, A.: The complexity of the Clar number problem and an exact algorithm. J Math Chem **56**, 597–605 (2018). https://doi.org/10.1007/s10910-017-0799-8
8. Caporossi, G., Hansen, P.: Enumeration of polyhex hydrocarbons to h = 21. Journal of Chemical Information and Computer Sciences **38**(4), 610–619 (1998). https://doi.org/10.1021/ci970116n, https://doi.org/10.1021/ci970116n
9. Clar, E.: The Aromatic Sextet. Wiley (1972)
10. Cyvin, J., Brunvoll, J., Cyvin, B.N.: Search for Concealed Non-Kekuliian Benzenoids and Coronoids. J. Chem. Inf. Comput. Sci. **29**(4), 237 (1989)
11. Devriendt, J., Bogaerts, B., Bruynooghe, M., Denecker, M.: Improved static symmetry breaking for sat. In: Creignou, N., Le Berre, D. (eds.) Theory and Applications of Satisfiability Testing – SAT 2016. pp. 104–122 (2016)
12. Di Giovannantonio, M., Yao, X., Eimre, K., Urgel, J.I., Ruffieux, P., Pignedoli, C.A., Müllen, K., Fasel, R., Narita, A.: Large-Cavity Coronoids with Different Inner and Outer Edge Structures. Journal of the American Chemical Society **142**(28), 12046–12050 (2020). https://doi.org/10.1021/jacs.0c05268
13. Dias, J.R.: 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). https://doi.org/10.1021/jp806987f
14. Draine, B.T.: Astronomical Models of PAHs and Dust. EAS Publications Series **46**, 29–42 (2011). https://doi.org/10.1051/eas/1146003
15. Fages, J.G., Lorca, X., Prud'homme, C.: Choco solver user guide documentation. https://choco-solver.readthedocs.io/en/latest/
16. Ismail, I., Stuttaford-Fowler, H.B.V.A., Ochan Ashok, C., Robertson, C., Habershon, S.: Automatic Proposal of Multistep Reaction Mechanisms using a Graph-Driven Search. The Journal of Physical Chemistry A **123**(15), 3407–3417 (2019). https://doi.org/10.1021/acs.jpca.9b01014

17. Kim, Y., Kim, J.W., Kim, Z., Kim, W.Y.: Efficient prediction of reaction paths through molecular graph and reaction network analysis. Chemical Science **9**(4), 825–835 (2018). https://doi.org/10.1039/C7SC03628K
18. Luch, A.: The Carcinogenic Effects of Polycyclic Aromatic Hydrocarbons. Imperial College Press, London (2005), https://www.worldscientific.com/worldscibooks/10.1142/p306
19. Mann, M., Nahar, F., Schnorr, N., Backofen, R., Stadler, P.F., Flamm, C.: Atom mapping with constraint programming. Algorithms for Molecular Biology **9**(1), 23 (2014). https://doi.org/10.1186/s13015-014-0023-3
20. Mishra, S., Beyer, D., Eimre, K., Kezilebieke, S., Berger, R., Gröning, O., Pignedoli, C.A., Müllen, K., Liljeroth, P., Ruffieux, P., Feng, X., Fasel, R.: Topological frustration induces unconventional magnetism in a nanographene. Nature Nanotechnology **15**(1), 22–28 (2020). https://doi.org/10.1038/s41565-019-0577-9
21. Mishra, S., Beyer, D., Eimre, K., Liu, J., Berger, R., Gröning, O., Pignedoli, C.A., Müllen, K., Fasel, R., Feng, X., Ruffieux, P.: Synthesis and Characterization of $\pi$-Extended Triangulene. Journal of the American Chemical Society **141**(27), 10621–10625 (2019). https://doi.org/10.1021/jacs.9b05319
22. Narita, A., Wang, X.Y., Feng, X., Müllen, K.: New advances in nanographene chemistry. Chemical Society Reviews **44**(18), 6616–6643 (2015). https://doi.org/10.1039/C5CS00183H
23. Ricca, A., Bauschlicher, C.W., Boersma, C., Tielens, A.G.G.M., Allamandola, L.J.: The Infrared spectroscopy of compact polycyclic aromatic hydrocarbons containing up to 384 carbons. The Astrophysical Journal **754**(1), 75 (2012). https://doi.org/10.1088/0004-637X/754/1/75
24. Ricca, A., Roser, J.E., Peeters, E., Boersma, C.: Polycyclic Aromatic Hydrocarbons with Armchair Edges: Potential Emitters in Class B Sources. The Astrophysical Journal **882**(1), 56 (2019). https://doi.org/10.3847/1538-4357/ab3124
25. Rieger, R., Müllen, K.: Forever young: Polycyclic aromatic hydrocarbons as model cases for structural and optical studies. Journal of Physical Organic Chemistry **23**(4), 315–325 (2010). https://doi.org/10.1002/poc.1644
26. Simoncini, D., Allouche, D., de Givry, S., Delmas, C., Barbe, S., Schiex, T.: Guaranteed Discrete Energy Optimization on Large Protein Design Problems. Journal of Chemical Theory and Computation **11**(12), 5980–5989 (2015). https://doi.org/10.1021/acs.jctc.5b00594
27. Trinquier, G., Malrieu, J.P.: Predicting the Open-Shell Character of Polycyclic Hydrocarbons in Terms of Clar Sextets. The Journal of Physical Chemistry A **122**(4), 1088–1103 (2018). https://doi.org/10.1021/acs.jpca.7b11095
28. Wu, C.W.: Modelling Chemical Reactions Using Constraint Programming and Molecular Graphs. In: Principles and Practice of Constraint Programming. pp. 808–808 (2004)
29. Wu, J., Pisula, W., Müllen, K.: Graphenes as Potential Material for Electronics. Chemical Reviews **107**(3), 718–747 (2007). https://doi.org/10.1021/cr068010r