Journal of Mathematical Chemistry, Vol. 42, No. 4, November 2007 (© 2006)
DOI: 10.1007/s10910-006-9148-z

Fusenes and benzenoids with perfect matchings

Gunnar Brinkmann*

Applied Mathematics and Computer Science, University of Ghent, Krijgslaan 281 S9, B 9000 Ghent Belgium
E-mail: Gunnar.Brinkmann@Ugent.be

Christian Grothaus

Fakultät für Mathematik, Universität Bielefeld, D 33501 Bielefeld, Germany

Ivan Gutman

Faculty of Science, University of Kragujevac, P.O. Box 60, 34000 Kragujevac, Serbia & Montenegro

Received 24 April 2006; revised 15 May 2006

In this paper, a method to constructively enumerate fusenes and benzenoids with perfect matchings is given. It is based on an algorithm for generating all fusenes and benzenoids and introduces new restrictions into the generation process that avoid the generation of structures without perfect matchings.

KEY WORDS: benzenoid, fusene, Kekulé structure, matching, enumeration

AMS subject classification: 92E10, 68R10, 05A15, 05C30, 05C70, 05C85

1. Introduction

A fusene is a conjugated molecule composed of mutually condensed six-membered rings [1]. Fusenes are divided into benzenoid and helicenoid systems.

In the mathematical abstraction, a *fusene* is a simple planar 2-connected graph embedded in the plane with all the vertices of degree 2 or 3, all bounded faces hexagons and all vertices not in the boundary of the outer face of degree 3. In this language benzenoids are fusenes that are isomorphic to subgraphs of the hexagonal lattice, whereas helicenoids are fusenes that are not subgraphs of the hexagonal lattice.

For an introduction into the mathematical and chemical properties of these structures see [2] or [3].

Due to their importance there is a long history of algorithms for enumerating benzenoids and fusenes, see e.g. [1,4–20].

*Corresponding author.

All the algorithms in these articles are of a constructive nature, that is: they do not only determine the number of structures, but they are also able to output one element of every isomorphism class (except for [20] where isomorphic structures are constructed and only the numbers of nonisomorphic ones are determined). In [21] a nonconstructive method to determine the numbers of isomers is given. On one hand this has the advantage of being able to compute the numbers of benzenoids for numbers of hexagons that are far out of reach for constructive enumeration methods, since for these algorithms the number of structures induce trivial lower bounds on the number of computer cycles needed to generate them – at least when one applies a more practically oriented definition of when one considers a structure to be *constructed*. But on the other hand, nonconstructive methods do not give the structures, so they do e.g. not allow to determine the energetically best structure or search for structures with certain properties. So for large numbers of hexagons it is a vicious circle: constructive enumeration is necessary in order to be able to examine the structures, but the constructiveness makes it impossible to generate all structures. In the case of fusenes the constructive generation is indeed very fast, but running tests on the structures – even very simple ones that need only linear time – is already much too time consuming for e.g. 26 hexagons. The only way out is to restrict the class under consideration in a way that all *interesting* structures are still contained. This is a challenge on the chemical side (which restrictions do not delete interesting structures) as well as on the mathematical side (restrictions are sometimes very hard to include in an efficient way – in some very difficult cases there are even no better methods known than constructing and filtering all structures ...).

As well known in the theory of benzenoid hydrocarbons [3, 22–25], benzenoid hydrocarbons are chemically stable only if their molecular graphs have perfect matchings (or, in the language of chemistry: if they have Kekulé structures, i.e., if they are Kekuléan).

One should note that the existence of perfect matchings is a necessary, but not a sufficient condition for chemical stability. The fact is that there exist highly unstable Kekuléan benzenoids [24,26], but there do not exist stable non-Kekuléan species.

Anyway, the nonexistence of perfect matchings seems to be a reasonable criterion to avoid the computer-generation of the respective structures. In this article we will therefore describe how to efficiently generate benzenoids and fusenes with perfect matchings. We will do this by including the restriction in an early step of the generation algorithm described in [7]. The results of our computations are given in tables 1 and 2.

In order to understand this approach it is not necessary to know all the details of the algorithm in [7] – especially not the methods used for isomorphism rejection and the details of restricting the generation to benzenoids (which is done by a filter) – nevertheless it is necessary to know the basic construction principles, which we will describe now.

2. The algorithm

The following basic concepts and results were already given in [7], but are essential for understanding the new concepts presented here, that we will repeat them: The *inner dual* of a graph embedded in the plane is its dual graph with the vertex corresponding to the unbounded region removed. So it is a subgraph of the dual graph. While there is a one-to-one correspondence between embedded graphs and their duals, this is not the case for inner duals as can be seen in figure 1.

The inner duals of fusenes (which will be called *id-fusenes* in the remainder of this paper) are simple graphs. This can be shown directly, but is also a consequence of the results about boundary lengths in [27] and for a more general case in [28].

A *labelled embedded graph* is an embedded graph together with a map from the set of all *angles* (that is pairs (e, e') with e' following e in the cyclic order of edges around their common start vertex) into the set of natural numbers. Two labelled embedded graphs are isomorphic if there is an isomorphism of the embedded graphs with the property that all angles are mapped onto angles with the same labels. The *labelled inner dual* of a planar graph is obtained by labelling every angle with the number of edges in the cyclic order of the dual graph between e and e'. Since the edges counted are in the dual, not the inner dual, the endpoints of edges that lie between two edges following each other in the rotational order of the inner dual, is the vertex corresponding to the unbounded face. If the graph has no bridges, so that the dual has no loops (especially no loops at the vertex corresponding to the outer face), we can reconstruct the dual of the planar graph from its labelled inner dual and the graph from its dual. So there is a one-to-one correspondence between bridgeless planar graphs and their labelled inner duals and two such graphs are isomorphic if and only if their labelled inner duals are (figure 2).

The approach described in [7] is in two steps: first all inner duals are constructed and then they are labelled in order to obtain fusenes. It is important for the approach presented here, that the number of fusenes is much bigger than the number of inner duals, e.g. by a factor of 70 for 13 hexagons, 850 for 20 hexagons and more than 7000 for 26 hexagons. This makes it efficient to still construct all inner duals and first filter them for those that cannot lead to fusenes with a perfect matching. A first easy lemma says:

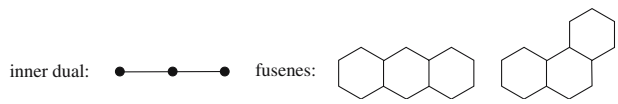

Figure 1. Two fusenes with the same inner dual.

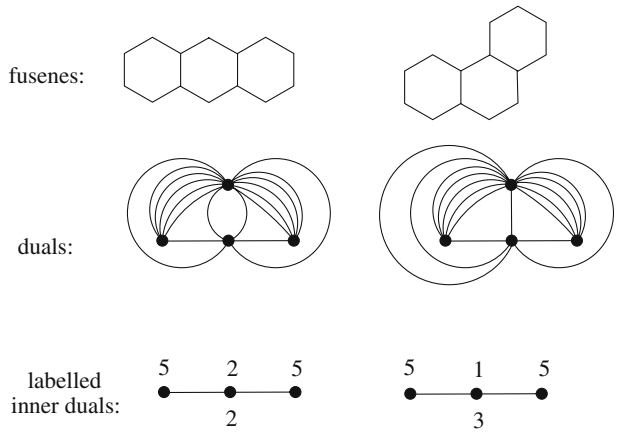

Figure 2. Two fusenes, their duals and labelled inner duals.

Lemma 1. *The number n of vertices of a fusene F is determined by its inner dual* $G = (V, E)$. *It is given by* $n = 5|V| - |E| + 1$

Proof. First note that as a consequence of the Euler formula the number of bounded triangles in G is $t = |E| - |V| + 1$. Since every vertex in G gives rise to a hexagon in F, the $|V|$ vertices in G give rise to $6|V|$ vertices in F. For every edge in G we have counted the endvertices of the corresponding edge in F twice, so we have to subtract $2|E|$ with the result that vertices corresponding to a triangle in G have been counted 3 times and subtracted 3 times – so once too often. Summing this up we have

$$n = 6|V| - 2|E| + t = 6|V| - 2|E| + |E| - |V| + 1 = 5|V| - |E| + 1$$

□

Since fusenes have to have an even number of vertices in order to have a perfect matching, this gives:

Corollary 1. *A fusene F with inner dual* $G = (V, E)$ *does not have a perfect matching if* $|V| + |E|$ *is even.*

This is a first and very efficiently computable criterium. Due to this criterium about 50% of the inner duals do not have corresponding fusenes with perfect matchings. So these inner duals can be rejected at once – leading to a speedup of 50%.

The remaining inner duals may or may not have labellings that lead to fusenes with perfect matchings. So the next step is to label the remaining inner duals in a way that the represented fusene has a perfect matching.

We refer the reader to [29] for details of the algorithm and the implementation. Here we will only give the basic idea of the algorithm and give some of the definitions in an a bit less formal way.

Fusenes are bipartite graphs and we always assume them to come with a two-colouring of the vertices (e.g. black and white) in a way that we never have two neighbouring vertices of the same colour. In the following we will repeatedly use the fact that in any matching you have an equal number of black and white saturated vertices.

Definition 1. Given an inner dual D of a fusene.

If v is a vertex in D, then $\mathrm{comp}(v)$ denotes the number of angles centered at v that bound the outer face.

A *block* of D is either a 2-edge-connected component of D, a vertex with degree 1 or a vertex v with $\mathrm{comp}(v) = 3$. A *fuseneblock* is the part of the fusene that is induced by faces corresponding to vertices in a block of the inner

Table 1
Numbers of fusenes and Kekuléan fusenes with up to 26 hexagons.

h	N	N_{kek}	$\frac{N_{\mathrm{kek}}}{N}$ (in %)
1	1	1	100.00
2	1	1	100.00
3	3	2	66.67
4	7	6	85.71
5	22	15	68.18
6	82	52	63.41
7	339	195	57.52
8	1505	807	53.62
9	7036	3513	49.93
10	33836	16025	47.36
11	166246	74848	45.02
12	829987	357602	43.09
13	4197273	1735566	41.35
14	21456444	8541951	39.81
15	110716585	42527067	38.41
16	576027737	213896060	37.13
17	3018986040	1085427249	35.95
18	15927330105	5552000102	34.86
19	84530870455	28600840970	33.83
20	451069339063	148280602424	32.87
21	2418927725532	773212222611	31.97
22	13030938290472	4053143014625	31.10
23	70492771581350	21348243725029	30.28
24	382816374644336	112936065636503	29.50
25	2086362209298079	599856634347474	28.75
26	11408580755666756	3197927731635661	28.03

Table 2
Numbers of benzenoids and Kekuléan benzenoids with up to 26 hexagons.

h	N	N_{kek}	$\frac{N_{\text{kek}}}{N}$ (in %)
1	1	1	100.00
2	1	1	100.00
3	3	2	66.67
4	7	6	85.71
5	22	15	68.18
6	81	51	62.96
7	331	190	57.40
8	1435	764	53.24
9	6505	3223	49.55
10	30086	14107	46.89
11	141229	62879	44.52
12	669584	284918	42.55
13	3198256	1304861	40.80
14	15367577	6031642	39.25
15	74207910	28082553	37.84
16	359863778	131573247	36.56
17	1751594643	619709418	35.38
18	8553649747	2932365897	34.28
19	41892642772	13931375631	33.25
20	205714411986	66422242556	32.29
21	1012565172403	317686257281	31.37
22	4994807695197	1523706173918	30.51
23	24687124900540	7326457682724	29.68
24	122238208783203	35307572182970	28.88
25	606269126076178	170501460668281	28.12
26	3011552839015720	824884145306863	27.39

dual. If B is a fuseneblock and M a matching, M_B denotes the set of edges that belong to M and B – so it is a matching of B. Note that even if M is a perfect matching, M_B need not be a perfect matching of B.

The components induced by vertices in the inner dual that are not contained in blocks can easily be seen to be paths. They are called *connection paths* and the corresponding subgraph of the fusene is called a *hexagonpath*. Again we write M_H for the edges of a matching M contained in a hexagonpath H.

A *connection edge* in an inner dual is an edge incident with a block, but not contained in it. A connection edge in a fusene is an edge that is contained in two different fuseneblocks or a fuseneblock and a hexagonpath (that is the dual of a connection edge in an inner dual).

The *decomposition graph* of a fusene is defined as follows: the set of vertices consists of all fuseneblocks and hexagonpaths and two vertices are adjacent if and only if the corresponding subgraphs of the fusene share an edge – which is by definition a connection edge. It is easy to see that the decomposition graph is a tree.

The blocks of an inner dual can be determined in linear time. For details see [29].

For every block the structure of the corresponding fuseneblock is uniquely determined – so also all possible parts of matchings that lie completely inside that part are already determined by the structure of the block. If the inner dual is 2-connected – so the whole inner dual is one big block, then there is one unique way how it must be labelled and the existence of a perfect matching is already determined by the inner dual. So in this case it all boils down to filtering (but note that – as mentioned in the introduction – in average there is a large and growing number of fusenes per inner dual).

Now assume that all angles belonging to vertices of blocks are already labelled. That means that though we do not know the whole structure of the fusene, we know at least the structure of the fuseneblocks and which are the *connection edges* – that is: edges they share with the rest of the fusene. Examples of blocks, fuseneblocks and connection edges are given in figure 3. A matching of the vertices of the fuseneblock or hexagon path can have the structures given in figure 4 on a connection edge. The structures denoted by D and D' can be dealt with together (and denoted by D) since whenever we have a matching that results in D we can just extend it to a matching that results in D' and the other way around: by removing an edge, D' can be reduced to form D. All the other types cannot be transformed into each other without affecting the interior vertices of the fuseneblock. If two fuseneblocks or a fuseneblock and a hexagon path share a connection edge, the type of the edge is interpreted differently depending on which part it is considered to be a subset of.

We will call a matching of a hexagonpath or fuseneblock *semiperfect* if it saturates (at least) all vertices not on connection edges.

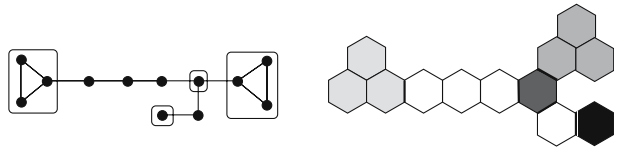

Figure 3. Blocks, fuseneblocks and connection edges.

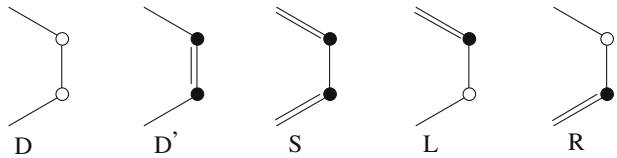

Figure 4. The possible configurations of saturation on the connection edges. The fuseneblock or hexagonpath is assumed to be on the left hand side in this picture.

Note 1. *Given a fusene with a perfect matching. If a connection edge is of type x when interpreted as part of one of the parts containing it and of type y when interpreted as part of the other, then the possibilities for (x, y) are (independent of order) (R, L), (D, D) or (D, S).*

This can be seen by just inspecting the cases.

Defining the type AD (standing for *any double*) for *any of type D or S* will simplify notation in the following lemmas and proofs:

Lemma 2. *Given a fusene F, a perfect matching M of F, a fuseneblock B and a connection edge $e \in B$.*

(i): *If e has type AD in M_B, then e has a type AD in M'_B for every perfect matching M' of F.*

(ii): *If e has type $t \in \{L, R\}$ in M_B, then e has the same type t in M'_B for every perfect matching M' of F.*

Proof. When looking at connection edges in fuseneblocks that are leafs of the decomposition tree (so fuseneblocks with just one connection edge) (i) and (ii) follow from counting the number of black and white vertices, which must be equal in case (i) and differ by one in case (ii), where the vertex from the smaller colour class must always be the saturated one. This restricts the possible structures of semiperfect matchings enough to prove (i) and (ii).

Now we can continue iteratively by always choosing an edge in a fuseneblock or hexagonpath that is a leaf of the subtree obtained when removing all the vertices of the decomposition tree that have already been considered. In such a component the types of all but at most one connection edge have already been proven to be determined by the lemma, so the same arguments can be used as for the case of a single connection edge. □

So let us now study how the types of the connection edges of hexagonpaths depend on the labels of the corresponding vertices in the inner dual. A labelsum of a connection path is the sum of all labels on one of the sides of the path. A connection path has two labelsums, but note that one is even (odd) if and only if the other is. The labelsum can also be interpreted as the number of edges in a path in the boundary of the corresponding hexagonpath that starts and ends in a vertex of a connection edge but doesn't contain one.

Lemma 3. (i): *For every hexagonpath and types $x, y \in \{L, R\}$ of the two connection edges e, e' some semiperfect matching exists that has the given types on the connection edges, if and only if the labelsum has a parity as in the following table:*

first type	*second type L*	*second type R*
L	*even*	*odd*
R	*odd*	*even*

(ii): *If the types of the connection edges are AD and D or both AD, there always exists a semiperfect matching with the given types on the connection edges – no matter what the labels or labelsums are.*

(iii): *If the given types of the connection edges are both D, there exists a semiperfect matching with the given types on the connecting edges if and only if one of the labels of the angles of the connecting path is odd.*

We do not consider the case of type *S* given for a connection edge of a hexagonpath, because in the remainder we won't need that case.

Proof. Note that an even/odd labelsum means that there is a path of even/odd length between the right vertex of one connection edge and the left of the other – and this again means that the right vertex has the same, resp. a different colour than the left vertex of the other.

First assume that a semiperfect matching is given. If case (i) applies and the first edge has only a white vertex saturated, the other edge has only a black vertex saturated. So in case the vertex is a left vertex (type L) and the labelsum is even, then the right vertex of the other edge has the same colour as the saturated vertex – so the other edge must have type L too. The other cases follow in the same way.

The existence of a semiperfect matching under the given conditions follows easily: all vertices are in one of the boundary paths of the hexagonpath for which the labelsum can be formed. So assume e.g. that the type of connection edge c_1 is L and that of c_2 is R and the labelsum is odd. Then take the boundary path $e_1, e_2, \ldots, e_k$ from the left vertex of c_1 to the right one of c_2 and choose every edge with an odd index for the matching. Since the path has an odd length, all vertices on the path are saturated. Then take the path $e_1, e_2, \ldots, e_m$ from the right vertex of c_1 to the left one of c_2 and choose every edge with an even index for the matching. Since the path has an odd length, all vertices on the path except the first and last are saturated, so we have a semiperfect matching of the given type. The other cases can be dealt with analogously.

Part (ii) follows immediately: since the boundary cycle $e_1, e_2, \ldots, e_k$ of a hexagonpath has even length, the edges with even index and the edges with odd index both form a perfect matching. If one edge is of type D, we can choose the matching containing that one, otherwise we can choose an arbitrary one.

Part (iii) can be proven using part (ii): For the case of just one hexagon we can check it directly. Now assume that you have more than one hexagon. Split the path into two nonempty paths P_1, P_2 with the edge along which it is cut the new connection edge of the parts. Assume the odd label is in part P_1. Then by induction there exists a semiperfect matching of P_1 with type D on both connection edges. But now any matching of P_2 with type AD on the new connection edge and D on the old can be used to construct the matching we were searching for. □

Now the basic strategy to proceed is obvious: Given an inner dual, we first compute the blocks and connection paths and label the angles belonging to block-vertices. Then for the blocks the semiperfect matchings are computed. If the block B has n connection edges $c_1, \ldots, c_n$, all possible n-tuples $(T_1, \ldots, T_n)$ are computed so that there exists a semiperfect matching of B where c_i has type T_i for $1 \leq i \leq n$. We call these the *type vectors*.

Figures 5 and 6 give an example. The components A,B,C are identical and since they have only six black vertices and seven white ones, it is immediate that – if at all – they can only have a semiperfect matching leaving the white vertex of the connection edge c_1 unmatched. It is easy to check that such a matching indeed exists, so the only possible type vector is (R). For the component D we have several possible combinations of types for the connection edges c_1, c_2, c_3 as given in the figure.

Then we compute all combinations of type vectors that can be combined in a way that the connection paths can be labelled in a way that semiperfect ma-

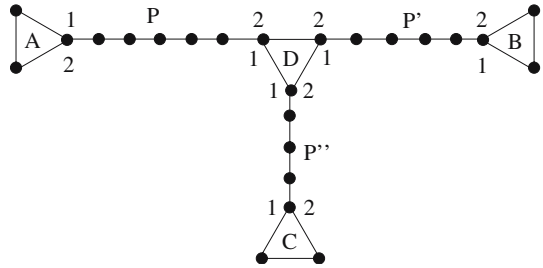

Figure 5. An example of a partially labelled inner dual.

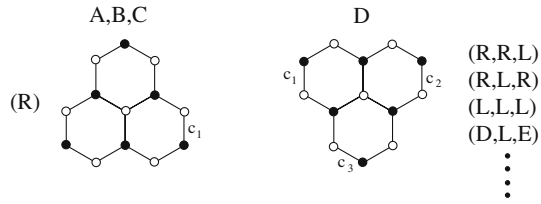

Figure 6. The type vectors of the blocks in figure 5.

things of the paths exist that can extend semiperfect matchings of the blocks to form a perfect matching of the fusene: Given two blocks connected by a connection path. If the connection path is empty, the types (independent of order) of the connection edge interpreted from the two blocks must be $\{L, R\}$ or $\{D, S\}$ or $\{D, D\}$. For a nonempty connection path the types of the two connection edges of the blocks must be $\{L, L\}, \{L, R\}, \{R, R\}, \{D, S\}, \{D, D\}$ or $\{S, S\}$.

Let us write o for *odd labelsum*, e for even labelsum, c for *must contain odd label* and a for *all labels possible*. If we have connection paths $P_1, \ldots, P_n$, we can now compute all *labelling vectors* $(L_1, \ldots, L_n)$, with each $L_i \in \{o, e, c, a\}$ telling us which combinations of labellings of the angles of the connection paths lead to Kekuléan structures. The value of L_i describes how P_i must be labelled. Then we can just do the labelling according to these rules and are guaranteed to produce on one hand only Kekuléan fusenes or benzenoids but on the other hand we are guaranteed to generate them all.

In the example the types (R) for blocks A, B, C and (R, R, L) for block D gives the labelling type (e, e, o) – the first and second connection path must be labelled with an even label sum and the third with an odd one. So we can start to label the first path in an arbitrary way – just choosing the last label in a way to make sure the sum is odd. For each labelling of the first path we can label the second path in an arbitrary way – except for the last vertex, etc. All possible labelling types for the example are (e, e, o), (e, o, e) and (o, o, o).

While the program our algorithm is based upon takes care of isomorphisms between fusenes with different labels, it would not detect the fact that the same labelling is produced twice. So our algorithm has to make sure that this does not happen. But if we had e.g. labelling possibilities (e, e, a) and (e, e, c), all labellings generated for (e, e, c) would be repeated when labelling according to the rule (e, e, a). But as we will see, the problem with a and c is the only one.

Lemma 4. *Given an inner dual of a fusene with connections paths* $P_1, \ldots, P_n$. *Then for every* $1 \leq i \leq n$ *the labelling type of* P_i *is either in* $\{e, o\}$ *for every labelling of the blockvertices and every possible labelling vector or in* $\{a, c\}$ *for every labelling of the blockvertices and every possible labelling vector.*

Proof. This result can be proven analogously to Lemma 2.: for connections paths neighbouring a leaf of the decomposition tree it follows immediately, since the number of vertices of the block that is a leaf is either even (then the type must be in $\{a, c\}$) or odd (then the type must be in $\{e, o\}$). This also implies whether an even or odd number of the vertices on connection edges must be saturated from inside the path. Now we can remove the leafs of the decomposition tree together with their neighbours. For the new leafs there is at most one connection edge for which the parity of the number of vertices that would be

matched from outside by a perfect matching is not yet determined. So again an easy parity argument gives us the result for the connection path starting at this edge. □

Now assume that for two different labelling vectors $(L_1, \ldots, L_n)$, $(L'_1, \ldots, L'_n)$, the same labelling of the inner dual is produced. If the two vectors differ in L_i, then obviously L_i cannot be in $\{e, o\}$, since labellings of these types are always different. So we must have $L_i = a$ and $L'_i = c$. To solve this problem we replace every labelling vector with $L_i = a$ by two labelling vectors – once with $L_i = c$ and once with $L_i = \bar{c}$ which is a new type saying that only even labels (this means: only label 2) may be used. After this replacing (and removing doubles obtained this way), no two labelling vectors produce the same labelling.

Of course the given description is just a simplified sketch. Some of the more technical details used to speed up the computation are not mentioned here. It is e.g. obvious that in some cases not *all* type vectors have to be computed. In the example we would first determine the type vector for the leafs of the decomposition tree. So when computing the type vector for D we already know that component A has type (R), then connection edge c_1 of component D must have type L or R – so it is of no use to compute type vectors with D or E for edge c_1 (and analogously for the other components). If a block with just one connection edge has type D on this edge (which is e.g. the often occuring case that it is a degree one vertex), the angles of the path can be labelled arbitrarily and this block and the connection path connecting it with the rest can simply be removed before computing the type vectors and combinations. In fact small blocks (like one vertex of degree one or three, three vertices, etc.) occur much more often than big ones – so for them one can simply precompute the type vectors instead of recomputing them again and again.

For details on the optimizations actually used in the program, see [29].

3. Testing

One of the most important tasks when implementing algorithms is to also develop testing routines. Each nontrivial algorithm bears the danger of numerous implementation errors – even when the implementation is done very carefully. Since the basic algorithm had been tested against various earlier results, it can be regarded as well tested and we only had to test the restriction to structures with perfect matchings. To this end we also implemented a very simple filter using Berge's path extension method. Using this filter we independently checked the numbers of benzenoids and fusenes with up to 21 hexagons. For more than 21 hexagons a complete check was not possible due to the large time consumption of this approach, so in order to increase the chance to detect possible errors that

can first occur for larger numbers of hexagons, we tested the operation only for certain (more or less) random *parts* of the set of all inner duals of fusenes/benzenoids with 22–30 hexagons. In all cases tested we had complete agreement of the results.

4. Results

The source code of the C-program based on this algorithm can be obtained from any of the authors. It can also compute some statistics about the structures generated (e.g. the numbers of structures grouped with respect to the automorphism group, the corresponding chemical formula or the combination of both).

Of course the ratio of Kekuléan structures among all structures does not only depend on the number of hexagons, but also on the chemical formula. If the formula is C_xH_y and x is odd, then all benzenoids and fusenes are obviously non-Kekuléan. If on the other hand the chemical formula is C_xH_y with $y = x/2+3$ then the structures are catacondensed and all vertices are in the boundary of even length – so all of them are Kekuléan. For h hexagons this is the case with $x = 4h+2$. The smallest possible number of atoms for a given number of hexagons is $2h + 1 + \lceil\sqrt{12h-3}\rceil$. This cannot as easily be determined as the largest one, so we refer the reader to [27] for a proof. In [28,30] an independent and more general proof and more general formulas are given that allow besides purely hexagonal structures also structures with up to six pentagons. One might expect that formulas that correspond to the *most pericondensed* structures – that is structures with the smallest possible number of atoms for a given number of hexagons – have the smallest ratio of Kekuléan structures among those with an even number of vertices, but as the following tables show this is at least for these small numbers of hexagons not the case. Using the number of C-atoms to measure the distance, the formula for which the minimum ratio of Kekuléan structures is obtained even seems to be closer to the catacondensed case (where we have the maximum number of C-atoms and provably a ratio of 100% Kekuléan structures) than to the formula with the minimum number of atoms. Example statistics for the numbers of Kekuléan and non-Kekuléan structures for 24 hexagons and all possible chemical formulas are given in tables 4 and 4. For other vertex numbers the tables can be obtained from any of the authors. It would be interesting to know for which formula with an even number of C-atoms the ratio is below 50% for the first time – if such a formula exists.

In the following tables N stands for the numbers of benzenoid or fusene isomers and N_{kek} for the number of all Kekuléan benzenoid or fusene isomers. In tables 4 and 4 we only list formulas with an even number of C-atoms, because for the other formulas the number of Kekuléan structures is of course 0. *Sum* stands for the sum of the above numbers – that is *all*, resp. *all Kekuléan* benze-

Table 3
Chemical formulas for fusenes with 24 hexagons.

Isomer	N	N_{kek}	$\frac{N_{\text{kek}}}{N}$ in %
$C_{66}H_{20}$	1	1	100.00
$C_{68}H_{22}$	789	503	63.75
$C_{70}H_{24}$	34324	21533	62.73
$C_{72}H_{26}$	648833	405106	62.44
$C_{74}H_{28}$	8716742	5357926	61.47
$C_{76}H_{30}$	96903418	58148416	60.01
$C_{78}H_{32}$	928636403	542511324	58.42
$C_{80}H_{34}$	7694135780	4379676357	56.92
$C_{82}H_{36}$	55001434461	30550285983	55.54
$C_{84}H_{38}$	334388076753	181856671720	54.38
$C_{86}H_{40}$	1693160752488	905906495329	53.50
$C_{88}H_{42}$	6918072102201	3666866010067	53.00
$C_{90}H_{44}$	21702739655309	11521055373464	53.09
$C_{92}H_{46}$	48377948880959	26171976233456	54.10
$C_{94}H_{48}$	65964838261853	37725883455683	57.19
$C_{96}H_{50}$	41010359983640	27086116956577	66.05
$C_{98}H_{52}$	5640868033058	5640868033058	100.00
Sum	191706106257012	112936065636503	58.91
Total	382816374644336	112936065636503	29.50

Table 4
Chemical formulas for benzenoids with 24 hexagons.

Isomer	N	N_{kek}	$\frac{N_{\text{kek}}}{N}$ in %
$C_{66}H_{20}$	1	1	100.00
$C_{68}H_{22}$	789	503	63.75
$C_{70}H_{24}$	34324	21533	62.73
$C_{72}H_{26}$	643859	401936	62.43
$C_{74}H_{28}$	8341018	5127143	61.47
$C_{76}H_{30}$	86809987	52093183	60.01
$C_{78}H_{32}$	762002033	444923847	58.39
$C_{80}H_{34}$	5696736534	3238185434	56.84
$C_{82}H_{36}$	36317312851	20129384006	55.43
$C_{84}H_{38}$	195126389101	105852666208	54.25
$C_{86}H_{40}$	867366149134	462897519666	53.37
$C_{88}H_{42}$	3096731853207	1638063669232	52.90
$C_{90}H_{44}$	8477771216467	4494079237785	53.01
$C_{92}H_{46}$	16519224289831	8926908845540	54.04
$C_{94}H_{48}$	19872186264006	11318393656829	56.96
$C_{96}H_{50}$	10871927816586	7119266659018	65.48
$C_{98}H_{52}$	1218239791106	1218239791106	100.00
Sum	61161445650834	35307572182970	57.73
Total	122238208783203	35307572182970	28.88

noids or fusenes with an even number of C-atoms. *Total* also takes isomers with an odd number of C-atoms into account.

5. Conclusions

We gave a first approach to restrict the generation of fusenes and benzenoids to a subset that has a higher probability to occur in nature. Though the numbers are considerably smaller than for all structures, there are still way too many of them to perform time consuming tests or generate structures for a larger number of hexagons. Now the chemists are asked to develop more restrictive criteria that can be included in the generation process in order to further reduce the number of structures.

A modification of this approach might be used to also determine the number of different Kekulé structures of the fusenes and benzenoids generated or generate only those with at least some given number of Kekulé structures.

An interesting question proposed by the results from this article is what the asymptotic ratio of Kekuléan fusenes or benzenoids among all such structures is. The numbers of hexagons that can be dealt with by constructive methods are too small to suggest well based conjectures.

In this context it would also be interesting to investigate which formulas (with an even number of vertices) minimize the ratio of Kekuléan structures for a given number of hexagons.

References

[1] A.T. Balaban and F. Harary, Tetrahedron 24 (1968) 2505.
[2] J.R. Dias, *Handbook of Polycyclic Hydrocarbons*. Part A: benzenoid hydrocarbons. (Elsevier, 1987).
[3] I. Gutman and S.J. Cyvin, *Introduction to the Theory of Benzenoid Hydrocarbons* (Springer Verlag, 1989).
[4] A.T. Balaban, J. Brunvoll, J. Cioslowski, B.N. Cyvin, S.J. Cyvin, I. Gutman, W.J. He, J.V. Knop, M. Kovačević, W.R. Müller, K. Szymanski, R.Tošić and N. Trinajstić, Z. Naturforsch. 42a (1987) 863.
[5] K. Balasubramanian, J.J. Kaufmann, W.S. Koski and A.T. Balaban, J. Comput. Chem. 1 (1983) 149.
[6] G. Brinkmann, G. Caporossi and P. Hansen, MATCH Commun. Math. Comput. Chem. 43 (2001) 133.
[7] G. Brinkmann, G. Caporossi and P. Hansen, J. Algorithms 45 (2002) 155.
[8] G. Brinkmann, G. Caporossi and P. Hansen, J. Chem. Inf. Comp. Sci. 43 (2003) 842.
[9] J. Brunvoll, B.N. Cyvin and S.J. Cyvin, Top. Curr. Chem. 162 (1992) 65.
[10] J. Brunvoll, S.J. Cyvin, B.N. Cyvin, Z. Fuji and X.F. Guo, Struct. Chem. 7 (1996) 119.
[11] G. Caporossi and P. Hansen, J. Chem. Inf. Comput. Sci. 38 (1998) 610.
[12] G. Caporossi, P. Hansen and M. Zheng, in *Discrete Mathematical Chemistry DIMACS Workshop March 23–25 1998*, pp. 63–78, 2000.
[13] B.N. Cyvin, S.J. Xiaofeng, S.J. Cyvin and F. Zhang, Chem. Phys. Lett. 188 (1992) 537.

[14] W.J. He, W.C. He, Q.X. Quang, J. Brunvoll and S.J. Cyvin, Z. Naturforsch. 43a (1988) 693.
[15] J.V. Knop, W.R. Müller, K. Szymanski and N. Trinajstić, J. Chem. Inf. Comput. Sci. 30 (1990) 159.
[16] W.R. Müller, K. Szymanski and J.V. Knop, Croat. Chem. Acta 62 (1989) 481.
[17] W.R. Müller, K. Szymanski, J.V. Knop, S. Nikolić and N. Trinajstić, J. Comput. Chem. 11 (1990) 223.
[18] S. Nikolić, N. Trinajstić, J.V. Knop, W.R. Müller and K. Szymanski, J. Math. Chem. 4 (1990) 357.
[19] I. Stojmenović, R. Tošić and R. Doroslovacki, in *Graph Theory, Proceedings of the sixth Yugoslav Seminar on Graph Theory*, pp. 189–198. 1985.
[20] R. Tošić, D. Mašulović, I. Stojmenović, J. Brunvoll, S.J. Cyvin and B.J. Cyvin, J. Chem. Inf. Comput. Sci. 35 (1995) 181.
[21] M. Vöge, A.J. Guttmann and I. Jensen, J. Chem. Inf. Comp. Sci. 42 (2002) 456.
[22] A.T. Balaban, J. Mol. Struct. (Theochem) 120 (1985) 117.
[23] E. Clar, *The Aromatic Sextet* (Wiley, London, 1972).
[24] S.J. Cyvin and I. Gutman, J. Serb. Chem. Soc. 53 (1988) 391.
[25] M. Randić, Chem. Rev. 103 (2003) 3449.
[26] J. Aihara, Phys. Chem. Chem. Phys. 1 (1999) 3193.
[27] F. Harary and H. Harborth, J. Comb., Inf. Syst. Sci. 1 (1976) 1.
[28] J. Bornhöft, G. Brinkmann and J. Greinus, Eur. J. Combinat. 24 (2003) 517.
[29] C. Grothaus, Konstruktion von Fusenen und Benzenoiden mit perfekten Matchings, 2005. Diplomarbeit, Universität Bielefeld.
[30] J. Greinus, Patches mit minimaler Randlänge, 2001. Diplomarbeit, Universiät Bielefeld.