Enumeration of Polyhex Hydrocarbons to $h = 17$

Ratko Tošić and Dragan Mašulović

Institute of Mathematics, University of Novi Sad, 21000 Novi Sad, Yugoslavia

Ivan Stojmenović*

Computer Science Department, University of Ottawa, Ottawa, Ontario K1N 9B4, Canada

Jon Brunvoll, Bjorg N. Cyvin, and Sven J. Cyvin

Division of Physical Chemistry, The University of Trondheim, N-7034 Trondheim-NTH, Norway

Received June 16, 1994[⊗]

This paper describes a rather efficient algorithm that enumerates nonisomorphic geometrically planar, simply connected polyhexes (hexagonal systems). It has been implemented in Modula-2 and used to determine and classify the pertinent systems with $h \leq 17$ hexagons. The result for $h = 17$, viz., 1751594643, is new.

INTRODUCTION

A classical paper on the enumeration of polyhex hydrocarbons[1] dates back to 1968. But it was not before 1983 that the Düsseldorf–Zagreb group (Knop and Trinajstić with collaborators), inspired by Balasubramanian et al.,[2] published their results[3] from computerized enumerations of geometrically planar, simply connected polyhexes to $h = 10$, where h is the number of hexagons. The results for $h = 11$ came from Novi Sad,[4] while the $h = 12$ data emerged from a Chinese–Norwegian collaboration.[5] Gutman[6] emphasized the difficulties in computations of this kind for large h values and proposed therefore an empirical function, which was supposed to reproduce approximate $N(h)$ numbers of the polyhexes under consideration. However, his formula, which was based on exact $N(h)$ numbers for $h \leq 11$, failed to predict $N(12)$ with a sufficient accuracy. Therefore, in 1988, Aboav and Gutman[7] refined the original formula and incorporated the exact values known at that time, viz., those for $h \leq 12$.

Now it was time for the Düsseldorf–Zagreb group to regain the hegemony in the computations of the $N(h)$ numbers of geometrically planar, simply connected polyhexes. They published[8] $N(13)$ in 1989 and shortly thereafter the numbers for[9] $h = 14$, for[10] $h = 15$, and[11] for $h = 16$. Müller et al.[8] deemed the prediction for $N(13)$ by Aboav and Gutman[7] to be unsatisfactory. We have come to a different conclusion in this matter (see below).

In the present work we achieved the computational results for $N(17)$. However, the breaking of record was not the only motivation of this work. On the other hand, this achievement is a clear indication of the efficiency of the computer algorithm which was developed, and the main purpose of this work is just a presentation of this algorithm. It exploits several ideas: it generates polyhexes in a limited region of hexagonal lattice called cage and it generates polyhexes by generating the boundary line and counting the boundary hexagons. The algorithm classifies the polyhex hydrocarbons according to their perimeter length, which was not done in the cases of $h > 11$ in the works cited above. The present algorithm uses the results of the enumeration and classification of polyhexes according to their various kinds of symmetry. These enumerations were performed by separate programs and are scheduled for future publications.

It may be argued that a ten-digit number like $N(17)$ is of no interest for chemists actually. However, exact numbers of this kind have proved to be very useful when it comes to the classification of polyhexes in different directions. Then they often break down to reasonably small numbers of considerable interest. The classification into symmetry groups[12] is one aspect of this feature. Another aspect is the enumeration of C_nH_s isomers, which is especially important from a chemical point of view. Dias[13] is the pioneer on the systematization of the chemical formulas (C_nH_s) of polyhexes and has also given significant contributions to their enumeration. It has been noted[14] that a classification of polyhexes according to the number of internal vertices[3] n_i, and according to the perimeter length[4] n_e, contain the information on C_nH_s isomers. In the present work, a combined classification into symmetry groups and C_nH_s isomers is reported.

BASIC DEFINITIONS

Polyhexes. A polyhex (system) is a connected system of congruent regular hexagons such that any two hexagons either share exactly one edge or they are disjoint.[9,11] Presently we shall be interested only in the class of geometrically planar, simply connected polyhexes. A polyhex is geometrically planar when it does not contain any overlapping edges, and a simply connected polyhex has no holes (see Figure 1). In this way the helicenes[15,16] and coronoids[17] (molecules with holes[18]) are excluded.

The geometrically planar, simply connected polyhexes have often been referred to as "benzenoids".[19–21] These systems may conveniently be defined in terms of a cycle on a hexagonal lattice; the system is found in the interior of this cycle, which represents the boundary (usually called the "perimeter") of the system. With the aim of avoiding confusion we have adopted the term "hexagonal system" (HS)[22] for a geometrically planar, simply connected polyhex.

Free and Fixed Hexagonal Systems. Following Redelmeier[23] we introduce the notion of free and fixed HSs.

[⊗] Abstract published in *Advance ACS Abstracts,* January 15, 1995.

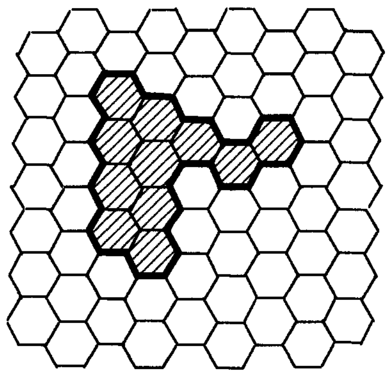

Figure 1. A geometrically planar, simply connected polyhex (hexagonal system) with $h = 11$ hexagons (hatched).

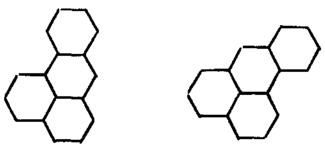

Figure 2. Two distinct fixed hexagonal systems (HSs), corresponding to one free HS.

Free HSs are considered distinct if they have different shapes (i.e., they are not congruent in the sense of Euclidean geometry). Their orientation and location in the plane is of no importance. For example, the two systems shown in Figure 2 represent the same free HS. Different free HSs are nonisomorphic. Fixed HSs are considered distinct if they have different shapes or orientations. Thus the two systems shown in Figure 2 are different fixed HSs. The key to the difference between fixed and free HSs lies in the symmetries of the HSs. Every free HS corresponds to 1, 2, 3, 4, 6, or 12 fixed HSs, depending on its symmetry. A HS is said to have a certain symmetry when it is invariant under the transformation(s) associated with that symmetry. Thus the HSs are classified into symmetry groups of which there are eight possibilities:[20,21] D_{6h}, C_{6h}, D_{3h}, C_{3h}, D_{2h}, C_{2h}, C_{2v}, and C_s. The number of fixed HSs for each free HS under these symmetry groups are specifically (in the same order) as follows: 1, 2, 2, 4, 3, 6, 6, and 12.

COMPUTER PROGRAMMING

Principles of Computation. Unfortunately there is no general formula for the number of nonisomorphic hexagonal systems with given number of hexagons. Therefore, one can only enumerate and classify them by using a brute force approach, fast computers, and algorithms. This paper presents one such algorithm. It has been used to enumerate nonisomorphic HSs with $h \leq 17$ hexagons and to classify them according to their perimeter.

In the present computation the symmetry of the HSs was exploited by adopting the method of Redelmeier.[23] This method was improved in some aspects by using a boundary code[4,24] for the HSs.

The exploitation of symmetry involves a separate enumeration of the fixed HSs on one hand and free HSs of specific symmetries on the other.

Let $H(h)$ be the number of fixed HSs with h hexagons, while $N(h)$ is used to denote the number of free (nonisomorphic) HSs with h hexagons. Furthermore, let $N(h)$ be split into the numbers for the different symmetries, say

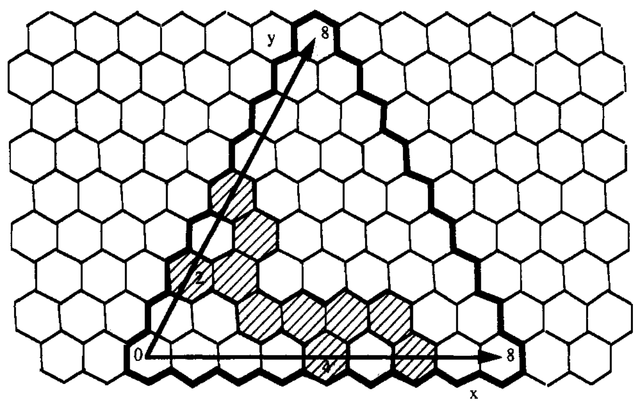

Figure 3. A HS placed in cage(9).

$N(\mathrm{G},h)$, where G indicates the symmetry group. Then

$$H(h) = N(D_{6h},h) + 2N(C_{6h},h) + 2N(D_{3h},h) + 4N(C_{3h},h) + 3N(D_{2h},h) + 6N(C_{2h},h) + 6N(C_{2v},h) + 12N(C_s,h)$$

For the free HSs,

$$N(h) = N(D_{6h},h) + N(C_{6h},h) + N(C_{6h},h) + N(D_{3h},h) + N(C_{3h},h) + N(D_{2h},h) + N(C_{2h},h) + N(C_{2v},h) + N(C_s,h)$$

Now, for a given h, the number $H(h)$ and the seven numbers $N(\mathrm{G}',h)$ are computed, where G′ should run through all the symmetry groups except C_s. Next, the above two equations are used to determine the two remaining unknowns, $N(C_s,h)$ and $N(h)$.

Known results on the enumeration and classification of HSs are surveyed elsewhere.[25-27] These surveys include classification according to HSs symmetries. In some cases these results suffice for our purpose, and in some other cases we obtained necessary results, often for far greater number of hexagons or perimeter than required, by separate programs that will be reported elsewhere.

Cages. The easiest way to handle a beast is to put it in a cage. This is exactly what the present algorithm does. Cage is a rather regular region of the hexagonal lattice in which we try to catch all relevant hexagonal systems. This algorithm uses a triangular cage, the triangle being equilateral. Let cage(l) denote triangular cage with l hexagons along each side. Figure 3 shows cage(9) and demonstrates how a coordinate system can be introduced in cage(l).

It is almost obvious that each hexagonal system can be placed in the cage in such a way that at least one of its hexagons is on the x-axis of the cage, and at least one of its hexagons is on y-axis of the cage. We say that such HSs are properly placed in the cage.

Let H be a free HS with h hexagons and let G_H be its symmetry group. It can easily be shown that H can be properly placed in cage(h) in exactly $|G_H|$ ways.

Thus we generate and enumerate all HSs that are properly placed in the cage. Since we know how many times symmetric HSs have appeared, we can determine easily the number of all hexagonal systems as explained above. It suffices to determine the number of all nonisomorphic HSs with a given numbers of hexagons, provided we have enumerated all symmetric hexagonal systems. By doing this, we avoid isomorphism tests which are considered to be the most time-consuming parts of similar algorithms.

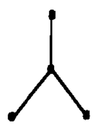

Figure 4. The central vertex in this configuration is said to be male.

Reducing the Cage. Cage(h) is supposed to catch all properly placed HSs with $\leq h$ hexagons. However, it turned out that the beasts are not that wild. Almost all hexagonal systems with h hexagons appear in cage($h - 1$). This results in significant speed up due to reduced search space. Those HSs that cannot be placed properly in cage($h - 1$) can easily be enumerated according to the following lemma (see also Table 1.

Lemma. The number of HSs with h hexagons which cannot be placed properly in cage($h - 1$) is $(h^2 - h + 4)2^{h-3}$. Among them there are $(h^2 - 3h + 2)2^{h-4}$ pericondensed (with exactly one internal vertex) and $(h^2 + h + 6)2^{h-4}$ catacondensed (i.e., with no internal vertex) HSs.

Proof. (a) Pericondensed HSs. Each vertex in the interior of cage(h) of the type male (see Figure 4) can be taken as the only internal vertex of some pericondensed HS properly placed in cage(h) but not in cage($h - 1$). The number of such vertices is $C(2,h - 1) = (S(h - 1,2)) = F((h - 1)(h - 2),2)$, and for each of them the number of HSs with h hexagons and with that vertex as the only inner vertex is 2^{h-3}. Hence it follows that the perimeter of all such pericondensed HSs is $4h$.

(b) Catacondensed HSs. (b1) Branched. Each of the $(S(h - 2,2))$ inner hexagons of cage(h) can be chosen as the only hexagon with three adjoint hexagons. For any of $(S(h - 2,2))$ choices of branching hexagon there are 2^{h-3} catacondensed HSs with h hexagons properly placed in cage(h) but not in cage($h - 1$). So, there are $(S(h - 2,2))2^{h-3}$ such HSs in total.

(b2) Unbranched. It is easy to see that there are $3 \cdot 2^{h-1} - 3$ unbranched catacondensed HSs containing the corner hexagon of cage(h) and $3(1 \cdot 2^0 + 2 \cdot 2^1 + 3 \cdot 2^2 + ... + (h - 3)2^{h-4}) = 3(h - 4)2^{h-3} + 3$ those without any corner hexagon. So, there are $3h \cdot 2^{h-3}$ unbranched catacondensed HSs which can be properly placed in cage(h) but not in cage($h - 1$). In total, there are $(h^2 + h + 6)2^{h-4}$ such catacondensed HSs, all of them having perimeter $2h + 2$.

The lemma follows by summing the numbers of HSs in both cases.

Therefore, we are allowed to use cage($h - 1$) when dealing with hexagonal systems with h hexagons.

Table 1 enumerates all occurrences of each HS, not only occurrences of nonisomorphic HSs. For $h = 3$ there are really 10 hexagons that are properly placed (i.e., have at least one hexagon on the x-axis and at least one hexagon on the y-axis), but manage to escape from cage(2) by at least one hexagon.

The Algorithm. The algorithm for enumerating HSs can be, on the highest level, described as follows:

1. initialize cage($h - 1$) and
2. generate and enumerate all hexagonal systems with h hexagons that are properly placed in the cage.

The rest of this section is devoted to accelerations and optimizations of this idea.

Let p and q be the smallest x- and y-coordinates (respectively) of all hexagons of a HS when the HS is properly placed in cage($h - 1$). Hexagons with coordinates $(p,0)$ and $(0,q)$ (with respect to the coordinate system of the cage) shall be of great importance to us. We shall name them key hexagons. Therefore, key hexagon on the x-axis has coordinate p, while key hexagon on the y-axis has coordinate q. Let H(p,q) denote the set of all HSs with $\leq h$ hexagons satisfying the following two conditions:

– it is properly placed in cage($h - 1$)

– its key hexagons on the x- and y-axes have coordinates p and q, respectively.

Table 1. Numbers of HSs with h Hexagons Which Cannot Be Properly Placed in Cage($h-1$) for $h \leq 18$

h	pericondensed	catacondensed	total
1	0	1	1
2	0	3	3
3	1	9	10
4	6	26	32
5	24	72	96
6	80	192	272
7	240	496	736
8	672	1248	1920
9	1792	3072	4864
10	4608	7424	12032
11	11520	17664	29184
12	28160	41472	69632
13	67584	96256	163840
14	159744	221184	380928
15	372736	503808	876544
16	860160	1130496	1990656
17	1966080	2555904	4521984
18	4456448	5701632	10158080

Figure 3 shows one element of H(4,2).

The family $\{H(p,q): 0 \leq p \leq h - 2, 0 \leq q \leq h - 2\}$ is a partition of the set of all hexagonal systems that are well placed in cage($h - 1$). Because of the overall beauty of symmetry, it can be verified that $|H(p,q)| = |H(q,p)|$, for all $p, q \in \{0,1,..., h - 2\}$. Thus the job of enumeration of all properly placed hexagons has reduced to determining $|H(p,q)|$ for all $p \geq q$. This gives the second iteration of the algorithm. It is not exactly what we have done, but is very close to it.

```
initialize Cage($h$-1);
total := 0;
for $q$:= 0 to $h$-2 do
    for $p$ := $q$ to $h$-2 do
        determine H($p,q$);
        $n$ := |H($p,q$)|;
        if $p$=$q$ then total := total +$n$ else total := total + 2$n$  fi
    od
od
```

Given the numbers $0 \leq p \leq q \leq h - 2$ and cage($h - 1$), determining H(p,q) reduces to generating all hexagons from H(p,q). We do that by generating their boundary line. A quick glance at Figure 3 reveals that the boundary line of a hexagonal system can be divided into two parts: the left part of the boundary (from the reader's point of view) which starts on the y-axis below the key hexagon and finishes at the first junction with x-axis and the rest of the boundary which we call the right part of the boundary.

We recursively generate the left part of the boundary line. As soon as it reaches the x-axis, we start generating the right part. All the time we take care of the length of the boundary line as well as the area of the hexagonal system. The trick which gives the area of the hexagonal system is simple: hexagons are counted in each row separately, starting from y axis, such that their number is determined by their x-coordinate. Each time the boundary goes up (down), we add (subtract, respectively) the corresponding x-coordinate. It is easy to see that, by following the contour of HS in counterclockwise direction (i.e., in the direction of generating HS, see Figure 3), there remain some hexagons out of HS to the left of the vertical contour line which goes down, while hexagons to the left of the vertical line which goes up belong to the HS. The "zigzag" movements do not interfere the area. Once the generating is over, the area of the HS gives the number of hexagons circumscribed in this manner. We have to do this because of useless hexagonal systems. Namely, during the generation of systems with h hexagons that belong to H(p,q) systems with $>h$ hexagons also appear, so we have to eliminate them.

Truly speaking, it would be a waste of time (and computing power) to insist on generating elements of H(p,q) strictly. This would require additional tests to decide whether the left part of the boundary has reached x-axis precisely at hexagon p or not. On the other hand, once we find out we have reached hexagon, say, $h + 2$, why should we ignore to find H($p + 2,q$)? This is why we are going to introduce another partition of the set of all properly placed HSs.

Given h and cage($h - 1$), put $H^*(q) = \cup^{h-2}{}_{j=0}\, H(j,q)$, for all $q = 0,1,...,h - 2$. It is obvious that $\{H^*(q)\colon\ 0 \le q \le h - 2\}$ is a partition of the set of all HSs with h hexagons that are properly placed in cage($h - 1$). Instead of having two separate phases (generating H(p,q) and adding appropriate number to total), we now have one phase in which generating and counting are put together. All we have to take care of is to prevent appearances of hexagonal systems from H(p,q) with $p < q$. But this causes no overhead, because it can be achieved by forbidding some left and some down turns in the matrix representing the cage. On the contrary, introducing the forbidden turns accelerates the process of generating the boundary line. Having all this in mind, the third iteration of the algorithm can be formulated as follows.

As we can see, the algorithm is a school example of backtracking, thus facing all classical problems of the technique: even for small values of h the search tree misbehaves so it is essential to cut it as much as possible. One idea that cuts some edges of the tree is based on the fact that for larger values of q there are some parts of the cage that cannot be reached by hexagonal system with $\le h$ hexagons but can easily be reached by useless HSs that emerge as side effect. That is why we can, knowing q, forbid some regions of the cage.

The other idea that reduces the search tree is counting the boundary hexagons. Boundary hexagon is a hexagon which has at least one side in common with the boundary line and which is in the interior of the hexagonal system we are generating. It is obvious that boundary hexagons shall be part of the HS, so we keep track on their number. We use that number as a very good criterion for cutting off useless edges in the search tree. The idea is simple: a further expansion of left/right part of the boundary line is possible if there are less than h boundary hexagons the line has passed by.

```
procedure ExpandRightPart(ActualPos);
begin
   if EndOfRightPart then
         n:= NoOfHexagons();
         if n≤h then
               determine p;
               if p=q then
                     total[n] := total[n] +1
               else
                     total[n] := total[n] +2
               fi
         fi
   else
         FindPossible(ActualPos, FuturePos);
         while RightPartCanBeExpanded ( ActuallPos, FuturePos) do
               ExpandRightPart( FuturePos);
               CalcNewFuturePos (ActualPos, FuturePos);
         od
   fi
end;
procedure ExpandLeftPart(ActualPos);
begin
   if EndOfLeftPart then
         ExpandRightPart (RightInitPos(q))
   else
         FindPossible (ActualPos, FuturePos);
           while LeftPartCanBeExpanded (ActualPos, FuturePos) do
                 ExpandLeftPart (FuturePos);
                 CalcNewFuturePos (ActualPos, FuturePos);
           od
     fi
  end;
  begin
     initialize Cage(h-1);
     set total[1..h] to 0;
     for q :=0 to h-2 do
           initialize y-axis key hexagon(q);
           ExpandLeftPart( LeftInitPos(q))
     od
  end.
```

The third idea that speeds up the algorithm is living on credit. When we start generating the left part of the boundary, we do not know where exactly is it going to finish on the x-axis, but we know that it is going to finish on the x-axis. In other words, knowing that there is one hexagon on the x-axis that is going to become a part of the HS, we can count it as a boundary hexagon in advance. It represents a credit of the hexagonal bank which is very eagerly exploited. Thus, many useless HSs are discarded before the left part of the boundary lands on the x-axis.

All these ideas collected in one place represent the core of the algorithm.

```
procedure ExpandRightPart(ActualPos, BdrHexgns);
begin
    if EndOfRightPart then
            n:= NoOfHexagons();
            if n≤h then
                    determine p;
                    if p=q then
                            total[n] := total[n] +1
                    else
                            total[n] := total[n] +2
                    fi
            fi
    else
            FindPossible(ActualPos, FuturePos);
            while RightPartCanBeExpanded ( ActuallPos, FuturePos)
            and BdrHexgns ≤h do
                    ExpandRightPart( FuturePos, update(BdrHexgns));
                    CalcNewFuturePos (ActualPos, FuturePos);
            od
    fi
end;
procedure ExpandLeftPart( ActualPos, BdrHexgns);
begin
    if EndOfLeftPart then
            ExpandRightPart (RightInitPos(q), updCredit(BdrHexgns))
    else
            FindPossible (ActualPos, FuturePos);
            while LeftPartCanBeExpanded (ActualPos, FuturePos)
            and BdrHexgns ≤ h do
                    ExpandLeftPart (FuturePos, update(BdrHexgns));
                    CalcNewFuturePos (ActualPos, FuturePos);
            od
    fi
end;
begin
    initialize Cage(h-1);
    set total[1..h] to 0;
    for q :=0 to h-2 do
            initialize y-axis key hexagon(q);
            ExpandLeftPart( LeftInitPos(q), InitBdrHexgns(q))
    od
end.
```

The Implementation. This paper describes only the main ideas in the algorithm. Many purely technical things had to be added in order to make a working computer program out of it.

The present algorithm has been implemented for IBM PC compatible computers in Modula-2. The program consists of five modules and more than 1900 bruto program lines. It has been used to determine the number of all properly placed HSs with $h \leq 17$ hexagons in cage(16).

Enumerating all properly placed hexagonal systems with 17 hexagons is a very lengthy process. Thus we had to divide the task into several smaller tasks; as a matter of fact, we had divided the enumeration process for $h = 17$ into 197 smaller tasks, which made it possible to run the program on several sites, namely Novi Sad, Ottawa, and Trondheim.

The parallelization was performed manually, and according to the natural criteria: the coordinate of the y-axis key hexagon and, since this was too coarse, according to the initial piece of the boundary. Instead of setting up the initial configuration for the cage and iterating on all possible values of q (which would yield a number of all hexagonal systems), this version of the program reads a configuration file, sets up the cage and other parameters according to information obtained from the file, and then calls the procedure to generate hexagonal systems. An example of a configuration file is shown below.

```
file name: H0601.INF
output 'H0601.DAT'
cage order = 16
base hexagon = 6
forbidden nodes:
  4,2 / 5,2 / 4,3 / 6,3 / 5,4 / 6,4 / 5,5 / 6,5 / 5,6
hex usage:
  3,2:4 / 4,3:2 / 5,4:1 / 5,5:3
generate left path:
   source left = 6,2 / came from: west
   source right = 6,6 / came from: west
hexagons = 4
area = -20
perimeter = 10.
```

RESULTS

The major achievement, which proves the great efficiency of our program, is the number of nonisomorphic hexagonal systems with $h = 17$, viz., $N(17) = 1751594643$. All the HSs with $h \leq 17$ are classified according to their perimeter lengths and their symmetries. Thus the present computations reproduced many numbers which were known;[25,26] a perfect consistency was observed throughout. But also a large amount of original results were produced, in fact too many to be listed here. An extract is given in the following.

The numbers of C_nH_s isomers of HSs are known completely for $h \leq 14$ according to a recent review.[26] These numbers had been classified in different ways, but most of them not according to symmetry. In Tables 2−4 we give the data of C_nH_s isomers for $h = 15-17$, respectively. They contain original total numbers for different isomers and are classified according to the symmetry groups. The numbers for C_s were omitted since they are readily obtainable as differences.

The coefficients (n, s) of the chemical formula (C_nH_s) are given by the relations $n = 4h - n_i + 2 = 2h + (n_e/2) + 1$ and $s = 2h - n_i + 4 = (n_e/2) + 3$, where the last expression is independent of h. From Tables 2−4 it is apparent that the perimeter length (n_e) is found in a certain domain, which depends on h. In precise terms, one has $2\lceil(12h-3)^{1/2}\rceil \leq n_e$

Table 2. Numbers of C_nH_s Isomers of HSs for $h = 15$, Classified According to Symmetry

n_e	n_i	formula	D_{3h}	C_{3h}	D_{2h}	C_{2h}	C_{2v}	total
62	0	$C_{62}H_{34}$	0	0	12	994	2860	4799205
60	1	$C_{61}H_{33}$	0	35	0	0	653	11642030
58	2	$C_{60}H_{32}$	0	0	0	1386	4279	16124124
56	3	$C_{59}H_{31}$	0	0	0	0	1083	15273424
54	4	$C_{58}H_{30}$	1	30	13	994	3107	11432430
52	5	$C_{57}H_{29}$	0	0	0	0	828	7215440
50	6	$C_{56}H_{28}$	0	0	8	457	1542	4077577
48	7	$C_{55}H_{27}$	0	21	0	0	418	2063017
46	8	$C_{54}H_{26}$	0	0	5	194	706	947291
44	9	$C_{53}H_{25}$	0	0	0	0	178	395860
42	10	$C_{52}H_{24}$	1	5	4	86	351	155656
40	11	$C_{51}H_{23}$	0	0	0	0	82	55857
38	12	$C_{50}H_{22}$	0	0	4	27	123	18396
36	13	$C_{49}H_{21}$	0	3	0	0	31	5612
34	14	$C_{48}H_{20}$	0	0	3	5	31	1570
32	15	$C_{47}H_{19}$	0	0	0	0	5	347
30	16	$C_{46}H_{18}$	1	1	1	2	12	70
28	17	$C_{45}H_{17}$	0	0	0	0	1	4
$h = 15$ (total)			3	95	50	4145	16290	74207910

Table 3. Numbers of C_nH_s Isomers of HSs for $h = 16$, Classified According to Symmetry

n_e	n_i	formula	D_{3h}	C_{3h}	D_{2h}	C_{2h}	C_{2v}	total
66	0	$C_{66}H_{36}$	4	55	13	3750	4670	18896981
64	1	$C_{65}H_{35}$	0	0	0	0	0	49231423
62	2	$C_{64}H_{34}$	0	0	5	5447	7036	72767011
60	3	$C_{63}H_{33}$	0	35	0	0	406	73855404
58	4	$C_{62}H_{32}$	0	0	26	4517	5918	58932191
56	5	$C_{61}H_{31}$	0	0	0	0	463	39460860
54	6	$C_{60}H_{30}$	5	16	13	2646	3575	23501017
52	7	$C_{59}H_{29}$	0	0	0	0	358	12575727
50	8	$C_{58}H_{28}$	0	0	7	1357	1872	6122480
48	9	$C_{57}H_{27}$	0	7	0	0	230	2721109
46	10	$C_{56}H_{26}$	0	0	7	581	829	1132642
44	1	$1C_{55}H_{25}$	0	0	0	0	137	436698
42	12	$C_{54}H_{24}$	1	4	5	215	323	156434
40	13	$C_{53}H_{23}$	0	0	0	0	58	51691
38	14	$C_{52}H_{22}$	0	0	0	80	121	16234
36	15	$C_{51}H_{21}$	1	1	0	0	28	4501
34	16	$C_{50}H_{20}$	0	0	2	20	31	1121
32	17	$C_{49}H_{19}$	0	0	0	0	9	223
30	18	$C_{48}H_{18}$	1	0	2	3	4	30
28	19	$C_{47}H_{17}$	0	0	0	0	1	1
$h = 16$ (total)			12	118	80	18616	26069	359863778

$\leq 4h + 2$, where $\lceil x \rceil$ denotes the smallest integer $\geq x$. The corresponding upper and lower bound for n_i has been reported several times.[19,21,26]

The overall distributions into symmetry groups for HSs are 25 known completely for $h \leq 16$ according to another recent review. Table 5 contains supplements to these data. For $h > 17$, the numbers for C_s are, of course, not known.

CONCLUSION

Turning back to the predictions of Aboav and Gutman[7] (see above), we are now able to subject their formula, viz., $N_{approx}(h + 1) = 4.98N(h)(1 - 5.77h^{-2})$, to a severe test. Notice approx that the formula predicts with confidence only one number $N(h + 1)$ when $N(h)$ is known exactly. Otherwise the errors accumulate too rapidly. The exact and predicted numbers in question are shown in Table 6 together with the errors defined in the same way as in the original paper.[7] The prediction for $h = 13$ is seen to be very good when judging from the percentage error. This conclusion is in variance with Müller et al.,[8] who apparently made a superficial judgment only. The percentage error increases when passing to $h = 14$ (cf. Table 6), but then it drops down again. The empirical observation[7] that the $N(h)$ numbers behave differently for odd or even h values is not confirmed. The prediction for $h = 17$ is seen to be excellent. On the basis of all this experience we are predicting with good confidence the $N(18)$ number of nonisomorphic HSs (with $h = 18$) to be $(8549 \pm 4) \times 10^7$.

So, are we going to start a computer calculation of $N(18)$? Definitely not. The above prediction does certainly not warrant such an endeavor, which at the end only would lead to one new predicted number. It seems to us that the present

Table 4. Numbers of C_nH_s Isomers of HSs for $h = 17$, Classified According to Symmetry

n_e	n_i	formula	D_{2h}	C_{2h}	C_{2v}	total
70	0	$C_{70}H_{38}$	14	3912	10656	74695032
68	1	$C_{69}H_{37}$	0	0	2400	208001048
66	2	$C_{68}H_{36}$	11	6122	17863	326681998
64	3	$C_{67}H_{35}$	0	0	4401	353437453
62	4	$C_{66}H_{34}$	19	5129	15208	299508694
60	5	$C_{65}H_{33}$	0	0	3925	212160087
58	6	$C_{64}H_{32}$	13	2748	8621	132831516
56	7	$C_{63}H_{31}$	0	0	2248	74811337
54	8	$C_{62}H_{30}$	10	1278	4254	38397592
52	9	$C_{61}H_{29}$	0	0	1087	18033634
50	10	$C_{60}H_{28}$	7	586	2152	7910833
48	11	$C_{59}H_{27}$	0	0	511	3235904
46	12	$C_{58}H_{26}$	11	229	915	1236839
44	13	$C_{57}H_{25}$	0	0	208	440491
42	14	$C_{56}H_{24}$	3	68	300	148430
40	15	$C_{55}H_{23}$	0	0	68	46166
38	16	$C_{54}H_{22}$	2	20	102	13286
36	17	$C_{53}H_{21}$	0	0	17	3414
34	18	$C_{52}H_{20}$	3	6	36	763
32	19	$C_{51}H_{19}$	0	0	6	117
30	20	$C_{50}H_{18}$	1	0	2	9
$h = 17$ (total)			94	20098	74980	1751594643

Table 5. Numbers of HSs Belonging to Different Symmetry Groups

h	D_{6h}	C_{6h}	D_{3h}	C_{3h}	D_{2h}	C_{2h}	C_{2v}
17	0	0	0	0	94	20098	74980
18	0	0	6	423	156	90265	120676
19	2	2	19	543	189	97913	348564
20	0	0	0	0	310	440230	563503
21	0	0	10	1923	365	479367	1635175
22	0	0	41	2507	615	2157946	2652445
23	0	0	0	0	748	2357108	7729807
24	0	0	16	8869	1237	10623852	12574028

Table 6. Exact and Approximate Numbers of HSs

h	$N(h)$	$N_{approx}(h)$	difference (error %)
1	1		
2	1		
3	3		
4	7	5	+2 (29)
5	22	22	0 (0)
6	81	84	−3 (3.7)
7	331	339	−8 (2.4)
8	1435	1454	−19 (1.3)
9	6505	6502	+3 (0.05)
10	30086	30087	−1 (0.03)
11	141229	141183	+46 (0.03)
12	669584	669782	−198 (0.03)
13	3198256	3200916	−266 (0.01)
14	15367577	15383524	−15947 (0.10)
15	74207910	74277568	−69658 (0.09)
16	359863778	360078349	−214571 (0.06)
17	1751594643	1751728873	−134230 (0.01)
18	unknown	8548784328	

work will be the last word in the overall enumerations of HSs. But the future is unforseeable. It may happen that some quite new principles come up for an efficient algorithm, which it would be worthwhile to test against computations of higher $N(h)$ numbers.

REFERENCES AND NOTES

(1) Balaban, A. T.; Harary, F. Chemical Graphs. V. Enumeration and Proposed Nomenclature of Benzenoid Cata-Condensed Polycyclic Aromatic Hydrocarbons. *Tetrahedron* **1968**, *24,* 2505–2516.

(2) Balasubramanian, K.; Kaufman, J. J.; Koski, W. S.; Balaban, A. T. Graph Theoretical Characterization and Computer Generation of Certain Carcinogenic Benzenoid Hydrocarbons and Identification of Bay Regions. *J. Comput. Chem.* **1980**, *1,* 149–157.

(3) Knop, J. V.; Szymanski, K.; Jeričević, O., Trinajstić, N. Computer Enumeration and Generation of Benzenoid Hydrocarbons and Identification of Bay Regions. *J. Comput. Chem.* **1983**, *4,* 23–32.

(4) Stojmenović, I.; Tošić, R.; Doroslovački, R. Generating and Counting Hexagonal Systems. In *Graph Theory, Proceedings of the Sixth Yugoslav Seminar on Graph Theory;* Dubrovnik, April 18–19, 1985; Tosić, R., Acketa, D., Petrović, V., Eds.; University of Novi Sad: Novi Sad, 1986; pp 189–198.

(5) He, W. J.; He, W. C.; Wang, Q. X.; Brunvoll, J.; Cyvin, S. J. Supplements to Enumeration of Benzenoid and Coronoid Hydrocarbons. *Z. Naturforsch.* **1988**, *43a,* 693–694.

(6) Gutman, I. Number of Benzenoid Hydrocarbons. *Z. Naturforsch.* **1986**, *41a,* 1089–1090.

(7) Aboav, D.; Gutman, I. Estimation of the Number of Benzenoid Hydrocarbons. *Chem. Phys. Lett.* **1988**, *148,* 90–92.

(8) Müller, W. R.; Szymanski, K.; Knop, J. V.; Nikolić, S.; Trinajstić, N. On Counting Polyhex Hydrocarbons. *Croat. Chem. Acta* **1989**, *62,* 481–483.

(9) Müller, W. R.; Szymanski, K.; Knop, J. V.; Nikolić, S.; Trinajstić, N. On the Enumeration and Generation of Polyhex Hydrocarbons. *J. Comput. Chem.* **1990**, *11,* 223–235.

(10) Nikolić, S.; Trinajstić, N.; Knop, J. V.; Müller, W. R.; Szymanski, K. On the Concept of the Weighted Spanning Tree of Dualist. *J. Math. Chem.* **1990**, *4,* 357–375.

(11) Knop. J. V.; Müller, W. R.; Szymanski, K.; Trinajstić, N. Use of Small Computers for Large Computations: Enumeration of Polyhex Hydrocarbons. *J. Chem. Inf. Comput. Sci.* **1990**, *30,* 159–160.

(12) Brunvoll, J.; Cyvin, B. N.; Cyvin, S. J. Enumeration and Classification of Benzenoid Hydrocarbons. 2. Symmetry and Regular Hexagonal Benzenoids. *J. Chem. Inf. Comput. Sci.* **1987**, *27,* 171–177.

(13) Dias, J. R. A Periodic Table for Polycyclic Aromatic Hydrocarbons. Isomer Enumeration of Fused Polycyclic Aromatic Hydrocarbons. 1. *J. Chem. Inf. Comput. Sci.* **1982**, *22,* 15–22.

(14) Brunvoll, J.; Cyvin, S. J. What do We Know about the Number of Benzenoid Isomers? *Z. Naturforsch.* **1990**, *45a,* 69–79.

(15) Randić, M.; Nikolić, S.; Trinajstić, N. Enumeration of Kekulé Structures for Helicenic Systems. *Croat. Chem. Acta.* **1988**, *61,* 821–831.

(16) Randić, M.; Gimarc, B. M.; Nikolić, S.; Trinajstić, N. On the Aromatic Stability of Helicenic Systems. *Gazz. Chim. Ital.* **1989**, *119,* 1–11.

(17) Brunvoll, J.; Cyvin, B. N.; Cyvin, S. J. Enumeration and Classification of Coronoid Hydrocarbons. *J. Chem. Inf. Comput.Sci.* **1987**, *27,* 14–21.

(18) Hall, G. G. Molecules with Holes. *Theor. Chim. Acta* **1988**, *73,* 425–435.

(19) Gutman, I. Topological Properties of Benzenoid Molecules. *Bull. Soc. Chim. Beograd* **1982**, *47,* 453–471.

(20) Cyvin, S. J.; Gutman, I. *Kekule Structures in Benzenoid Hydrocarbons; Lecture Notes in Chemistry 46*; Springer-Verlag: Berlin, 1988.

(21) Gutman, I.; Cyvin, S. J. *Introduction to the Theory of Benzenoid Hydrocarbons;* Springer-Verlag: Berlin, 1989.

(22) Sachs, H. Perfect Matchings in Hexagonal Systems. *Combinatorica* **1984**, *4,* 89–99.

(23) Redelmeier, D. H. Counting Polyominoes-Yet Another Attack. *Discrete Math.* **1981**, *36,* 191–203.

(24) Tošić, R.; Doroslovački, R.; Gutman, I. Topological Properties of Benzenoid Systems. XXXVIII, The Boundary Code. *MATCH* **1986**, *19,* 219–228.

(25) Cyvin, B. N.; Brunvoll, J.; Cyvin, S. J. Enumeration of Benzenoid Systems and Other Polyhexes. *Top. Curr. Chem.* **1992**, *162,* 65–180.

(26) Brunvoll, J.; Cyvin, B. N.; Cyvin, S. J. Benzenoid Chemical Isomers and Their Enumeration. *Top. Curr. Chem.* **1992**, *162,* 181–221.

(27) Balaban, A. T.; Brunvoll, J.; Cioslowski, J.; Cyvin, B. N.; Cyvin, S. J.; Gutman, I.; He, W. C..; He, W. J.; Knop, J. V.; Kovačević, M.; Müller, W. R.; Szymanski, K.; Tošić, R.; Trinajstić, N. Enumeration of Benzenoid and Coronoid Hydrocarbons. *Z. Naturforsch.* **1987**, *42a,* 863–870.

CI940076C