
COMPUTATIONAL GRAPH THEORY

K. BALASUBRAMANIAN*
Department of Chemistry, Arizona State University, Tempe, AZ 85287, USA

ABSTRACT

An important aspect of computational graph theory is the construction of algorithms and codes for graph-theoretical problems for which there are no simple analytical solutions, in general. There may be analytical solutions for certain particular graphs which could actually be used to debug the algorithms or codes. There are a number of graph-theoretical problems of chemical origin which require such algorithms suitable for computer codes due to the combinatorial complexities of such problems. Examples of such problems are matching polynomials of graphs, characteristic polynomials, chromatic polynomials, king, rook, and color polynomials, walk generating functions for graphs, spectral moment polynomials of graphs, etc. The objective of this chapter is to outline a few such problems and demonstrate that suitable algorithms when implemented on computers could provide powerful solutions for these problems. Applications outlined include bond structure of organic polymers and periodic structures, lattice statistics in statistical mechanics, computer simulation of NMR spectra, etc.

INTRODUCTION

Graph-theoretical techniques have a large number of chemical applications (ref. 1-42). These applications span a variety of areas such as quantum chemistry, spectroscopy, chemical kinetics, electronic structure of clusters, stereochemistry, structure elucidation, statistical mechanics, polymer chemistry and solid state chemistry. Graphs are useful representations of various topologically inequivalent arrangements of atoms in a molecule (isomers), quantum-mechanical interaction diagrams, graphical representation of statistical-mechanical integrals (Mayer-Mayer expansion), chemical reaction networks, NMR spin hamiltonians, etc.

Another intimately related field to graph theory is combinatorics which could be defined as a field of mathematics focusing on enumeration methods (number of ways of performing a given task). A number of chemical problems have served as motivating grounds for new techniques or theorems in combinatorics such as Pólya’s theorem which provides for a method to count configurations under group actions.

* Alfred P. Sloan Fellow; Camille and Henry Dreyfus Teacher-scholar
The objective of our chapter is to demonstrate the importance of computational algorithms and programs as aids in solving combinatorially complex graph theoretical problems of chemical origin. We show the use of recursive programming techniques for graph theoretical problems. Examples of problems which could be solved by such methods are finding matching polynomials of graphs, chromatic, king, color polynomials of graphs, symmetry groups of trees, etc. We also outline applications to band structure of polymers, exact lattice statistics in statistical mechanics and simulation of NMR spectra.

COMPUTATIONAL TECHNIQUES AND ALGORITHMS

A common computational technique employed in graph theory is recursion. In general, recursive method can be formulated as follows. Suppose G is a given graph. Let f be a function that we seek for the graph G (example: f could be a polynomial of some kind associated with the graph G). If f(G) can be expressed in terms of the f's associated with smaller graphs, then there exists a recursive relation for f. A recursive relation then takes the following form

\[ f(G) = P_1(x)f(G') + P_2(x)f(G'') + \ldots, \]

where f is a function of x, P_1(x), P_2(x) and known functions of x, f(G') and f(G'') are the corresponding functions for graphs G' and G'' etc., which are usually simpler than G itself (a vertex (vertices) or edge(s) less than G). A relationship is said to be recursive if both sides of the relationship contain the function that we seek although one side, in general, contains the same function for simpler graphs. There are a number of examples of recursive relations. A famous example is that of Fibonacci numbers which satisfy the following relation

\[ f_n = f_{n-1} + f_{n-2}, \]

where f_n's are integers. Another example of a recursive relation would be

\[ f(G_n) = x^2f(G_{n-1}) + f(G_{n-2}). \]

A recursive problem can be solved by an 'iterative method' if a starting point(s) (root(s) of the iteration) is known. For example, the Fibonacci numbers can be found iteratively since f_0 and f_1 can be defined. In the
following example, \( f_n \)'s can be found for any \( n \) iteratively since the roots \( f_0 \) and \( f_1 \) are known.

\[
f_0 = 1, \quad f_1 = 2
\]

\[
f_n = f_{n-1} + f_{n-2}
\]

If there is an iterative solution for a recursive problem one could solve it in a computational 'DOLOOP'. The algorithm would run as follows. An example of such an algorithm is shown below.

\[
f_0 + c_1
\]

\[
f_1 + c_2
\]

\[
\text{for } i = 2, n, \text{ DO}
\]

\[
f_i = f_{i-1} + f_{i-2}
\]

The output of the algorithm is a sequence of numbers \( \{f_0, f_1, f_2, \ldots, f_n\} \) which satisfy the recursive relation \( f_i = f_{i-1} + f_{i-2} \). Recursive relations for which roots are known are much simpler and easier to solve than the relations for which roots are not known and need to be found recursively.

Recursive relations for which roots are not known are very challenging, since they cannot be solved iteratively. An example of such a recursive relation is:

\[
f(G) = f(G-e) - f(G\theta e),
\]

where \( G-e \) is the graph obtained by deleting an edge \( e \) and \( G\theta e \) is the graph obtained by deleting the edge \( e \) and the vertices connected by \( e \), and all the edges connected to the vertices of \( e \). The roots of this recursion are not known, in general, since they differ for every graph. In this case, the only way to solve the recursive relation is by a recursive reduction of the graph in question until simple graphs are generated for which the function \( f \) can be easily found. In this case combinatorial explosions can occur very quickly even for relatively smaller graphs. Thus computers are needed to find the function for the graph \( G \) since it would be impossible to keep track of all the intermediate subgraphs which need to be reduced further.

Recursive reduction is graphically depicted by a tree shown in Fig. 1. In that tree the vertex \( s \) is the starting graph. Recursive reduction is applied
to s to generate the two vertices below it. Recursive reductions are applied to these vertices in turn etc., until the terminal vertices $T_1, T_2, \ldots T_n$ are reached for which say $f(T_1), f(T_2) \ldots f(T_n)$ are known. Then the $f$'s for the vertices above $T$'s are found etc. until the function for the starting vertex is obtained. These types of algorithms in general are combinatorially quite complex and require the use of computers.

![Recursive algorithm tree](image)

**Fig. 1.** Recursive algorithm tree.

**GRAPH POLYNOMIALS**

The matching polynomial $M_G$ of a graph $G$ is defined as

$$M_G(x) = \sum_{k=0}^{m} (-1)^k p(G,k) x^{N-2k}$$

where the nonadjacent numbers $p(G,k)$'s are defined as the number of ways in which $k$ mutually nonincident edges can be selected in a graph $G$ and $N$ is the number of vertices in the graph. Hosoya and co-workers (ref. 19,20) have
derived the basic recurrence relation to obtain the matching polynomial, namely

\[ M_G(x) = M_{G-e}(x) - M_{G\theta e}(x) \]

where \( G-e \) is a graph derived from \( G \) by deleting an edge \( e \) and \( G\theta e \) is the graph derived from \( G \) by deleting \( e \), the vertices of \( e \) and all the edges connected to the vertices of \( e \). The above relation is a recursive relation for which an iterative solution does not exist in general. Ramaraj and Balasubramanian (ref. 38) have developed a computer code to generate these polynomials. We review this method here.

The characteristic polynomial of a graph \( G \) is defined as the determinant \( |\lambda I - A| \), where \( A \) is the adjacency matrix of a graph which is defined as follows:

\[ A_{ij} = \begin{cases} 1 & \text{if the vertices } i \text{ and } j \text{ are connected} \\ 0 & \text{otherwise} \end{cases} \]

The problem of evaluating characteristic polynomials of graphs is considered a very difficult one. Balasubramanian (ref. 35,37) developed a computer program in Fortran for this problem. The graph \( G \) can be reduced successively until all resulting graphs are trees. Then we use the fact that matching polynomials are the same as characteristic polynomials for trees. We have used the program developed by Balasubramanian to compute the characteristic polynomials and assembled the matching polynomial of the initial graph \( G \) recursively. A computer program in Pascal based on the above algorithm was developed in reference 38. For details of this code, the readers are directed to this reference. We briefly illustrate the use of this code here. Fig. 2 shows the various steps executed in constructing the matching polynomial of naphthalene. Figure 3 shows a graph containing hexagons. The matching polynomial of this graph is shown below.

\[
\begin{align*}
x^{48} & - 60x^{46} + 1674x^{44} - 28850x^{42} + 344127x^{40} - 3016998x^{38} + 20152013x^{36} \\
& - 104913492x^{34} + 43196943x^{32} - 1419382254x^{30} + 3740060904x^{28} \\
& - 7914718788x^{26} + 13431639205x^{24} - 18200982024x^{22} + 19552772649x^{20} \\
& - 16479660654x^{18} + 10743316299x^{16} - 5315219724x^{14} + 1945680262x^{12} \\
& - 509172702x^{10} + 90806961x^8 - 10292946x^6 + 665136x^4 - 20328x^2 + 200
\end{align*}
\]
Fig. 2. Recursive steps in finding matching polynomials of naphthalene.

Another useful polynomial in chemical applications is the king polynomial which can be defined as a generator for the number of ways of placing non-taking kings on a chess board. Consider the chess board in Fig. 4. A king pattern is simply a way of placing kings (circles) on the chessboard so that no two kings can take each other (see Fig. 4). Let $C_k$ be the number of ways one can place $k$ non-taking kings on a chess board. Then the king polynomial can be defined as

$$K = 1 + C_1x + C_2x^2 + \ldots + C_kx^k + \ldots + C_nx^n$$
Fig. 4. A 2 x 3 chess board and king dimer and domino patterns on the chess board.

The king polynomial of a chess board G, $K_G(x)$ is recursively related to subgraph of G as (ref. 41)

$$K_G(x) = K_{G-C}(x) + x \cdot K_{G\mid C}(x),$$

where G-C is the chess board obtained by removing a cell C from G and G\mid C is the chess board resulting from the removal of the cell C and all the neighboring cells of C. Balasubramanian and Ramaraj (ref. 42) have shown that the graph in question can be recursively reduced until simple fragments (chains) are obtained. A Pascal code (ref. 42) was developed to compute king polynomial of any graph. A polynomial related to the king polynomial was defined by Balasubramanian and Ramaraj (ref. 42) which these authors call color polynomial. The color polynomial of a graph G is defined as

$$C = 1 + P_1x + P_2x^2 + \ldots + P_kx^k + \ldots + P_nx^n,$$
where \( P_k \) is the number of ways one can color the vertices of a graph with one type of color such that two adjacent vertices are not colored. A more general color polynomial \( C(x_1, x_2 \ldots x_n) \) is defined as follows.

\[
C(x_1, x_2, \ldots x_n) = \Sigma C(k_1, k_2 \ldots k_n) x_1^{k_1} x_2^{k_2} \ldots x_n^{k_n},
\]

where \( C(k_1, k_2 \ldots k_n) \) is the number of ways of coloring the vertices of a given graph with \( k_1 \) colors of the type 1 (e.g., blue), \( k_2 \) colors of the type 2 (e.g., red) etc., such that two adjacent vertices are not colored with the same color.

The color polynomial \( C \) satisfies the following recursive relation.

\[
C_G = C_{G-v} + x C_{G\bar{v}},
\]

where \( G-v \) is a graph obtained by removing a vertex \( v \) and \( G\bar{v} \) is the graph obtained by removing the vertex \( v \) and all the vertices which are adjacent to \( v \).

Rook polynomial of a chess board generates the number of ways of placing non-taking rooks. If \( r_k \) is this number then the rook polynomial \( R(x) \) is defined as

\[
R(x) = r_1x + r_2x^2 + \ldots + r_nx^n
\]

The rook polynomial satisfies the following recursive relation.

\[
R_G(x) = R_{G-c}(x) + x R_{G\bar{c}},
\]

where \( G-c \) and \( G\bar{c} \) have the same meaning as in the king polynomial relation. The rook polynomial of any chess board can be obtained using recursive reduction method.

The present author (ref. 34) showed that a recursive tree pruning method could be used to find the characteristic polynomials of trees. The tree whose characteristic polynomial is required is pruned until simple fragments are obtained. The characteristic polynomial of the given tree is then computed by assembling the polynomials of various fragments.

Recursive tree pruning technique could also be used to find the automorphism group of a given tree (ref. 33). The automorphism group of a tree can be expressed in terms of the groups of the pruned tree and the fragments resulting from pruning the tree. The pruning process is continued until a simple pruned tree and fragments are generated.
Characteristic polynomials of graphs which were defined earlier could be obtained using computational methods. The conventional determinant expansion is not a solution for this problem since the determinant of a matrix involves n! terms if n is the order of the matrix. A recursive matrix product algorithm formulated by Frame, (later called Leverrier-Faddeev method by Krivka et al. (ref. 40)) is another example of a recursive algorithm which provides for an elegant alternative solution (ref. 35,37) for a combinatorially complex problem. The present author (ref. 37) developed a computer code based on this method to show the power of this method. Characteristic polynomials of graphs containing a large number of vertices could be obtained easily.

APPLICATIONS

A. Organic polymers, lattices and walks.

Characteristic polynomials of organic polymers and periodic lattices were obtained by the present author (ref. 39) by generalizing the recursive algorithm outlined earlier to complex but hermetian matrices. Since characteristic polynomials of such lattices can be obtained relatively easily as opposed to their band structure these polynomials could serve as useful tools in many applications. Further self-returning walks could also be generated from these polynomials. Figure 5 shows the plot of the coefficient of the constant term of the characteristic polynomial of a polydiacytelene chain containing 61 unit cells as a function of n, where n is related to the reciprocal lattice vector.

B. Applications to exact finite lattice statistics.

In this section we briefly describe the application of color polynomials to lattice statistics. In the matrix approach to the grand canonical partition function of nearest-neighbor finite lattice systems (ref. 43,44), one treats the lattice within the cyclic boundary conditions. A square lattice is treated as a cylinder in this method. Any configuration of the lattice can be obtained in terms of molecules arranged in rows such that no two adjacent sites may be occupied (nearest neighbor exclusion). This corresponds to the enumeration of possible colorings of vertices arranged in a ring such that no two adjacent vertices carry the same color. One can then enumerate the number of compatible configurations of a given row with its adjacent row depending upon the nature of lattice (triangular, square, etc.) This gives rise to the matrix R described by Runnels and Combs (ref. 43).
Fig. 5. Plot of the coefficients of the constant term of the characteristic polynomial of a polydiacytelene chain containing 61 unit cells as a function of $n$ ($-N \leq n \leq N$), where $2N + 1$ is the total number of unit cells.

The largest eigenvalue of the matrix $R$ (denoted by $\lambda_1$) is critical in determining the thermodynamic properties of the associated lattice; for example, the pressure is given by:

$$\frac{p}{kT} = \frac{1}{M} \ln \lambda_1$$

where $M$ is the number of sites on any row in the cylinder. Also, the density is given by:

$$\rho = \left( \frac{z}{M \lambda_1} \right) \left( -\frac{d\lambda_1}{dz} \right),$$
where $z$ is the activity.

For a ring containing $n$ particles the color polynomial is given by:

$$C_n = 1 + g(n,1) x + g(n,2) x^2 + \ldots + g(n,k) x^k + \ldots + g(n,m) x^m$$

where $g(n,k) = \frac{n}{n-k} \binom{n-k}{k}$

g(n,k)'s are also known as menage numbers and they enumerate the number of ways of seating married couples at a round table such that the husband and wife do not sit next to each other.

C. Recursive tree pruning and Bethe lattices.

An interesting problem in statistical physics (ref. 45) is to enumerate number of ways of placing $k$ dimers on a lattice of $N$ points. The solution to this problem has got a number of applications especially in finding the grand canonical partition function of a lattice gas from which thermodynamic properties of the associated lattice could be obtained. 

Other applications include finding the partition function of a system of interacting magnets, adsorption of diatomics on metal surfaces and the enumeration of resonance structures.

Fisher and Essam (ref. 46) have shown the use of Bethe lattices for percolation and cluster size problems. While these are not true lattices, they are topological abstractions of true lattices which enable exact analytical evaluation of important physical properties. Bethe lattices are tree lattices. A Bethe lattice of valence $\sigma$ and length $n$ is defined as a tree in which each non-terminal vertex has $\sigma$ neighbors and there are $n$ bonds from the central vertex to any terminal vertex. To illustrate, Fig. 6 shows a Bethe lattice of valence 3 ($\sigma = 3$) and length 2 ($n = 2$). Recursive tree pruning method developed by the present author (ref. 34) for the characteristic polynomials of trees could be applied to Bethe lattices. The characteristic polynomial and the matching polynomial of a tree are the same. Since the matching polynomials generate the number of dimers on the associated lattices, the importance of these polynomials is obvious. To illustrate, the characteristic polynomial of a Bethe lattice of valence $\sigma$ and $n = 2$ is given by

$$\lambda^{(\sigma-1)^2} \cdot (\lambda^2 - \sigma+1)^{\sigma-1} \cdot (\lambda^2 - 2\sigma+1).$$

For $\sigma=3$ one obtains the characteristic polynomial of the Bethe lattice in Fig. 6 which is given by
\[ \lambda^4 (\lambda^2 - 2)^2 (\lambda^2 - 6 + 1). \]

Note that the above polynomial is factored in the tree-pruning method. Similar analytical expressions have been derived by the present author (ref. 47) for Bethe lattice with \( n = 3 \) and other \( n \) values.

![Bethe lattice diagram](image)

Fig. 6. A Bethe lattice of valence 3 and length 2.

D. NMR spectroscopy.

The present author (ref. 48) developed a computer code to enumerate the number of NMR signals (\(^{13}\text{C}, \ H \) or any nuclei) of any molecule. Recently, the present author (ref. 49) developed an algorithm and a computer code to construct the actual signal and intensity patterns. This code requires as input only the generators of the symmetry group of the molecule. As an illustration of the use of this algorithm and the computer code we show in Table 1, the \(^{13}\text{C} \) NMR signal and intensity patterns of linear polyacenes. Note that interesting patterns of NMR signals emerge for this class of compounds. Many such classes of polycyclic aromatics were used as illustrative examples by the present author (ref. 49) to demonstrate the power of these computational methods.
TABLE 1

$^{13}$C NMR signal and intensity patterns of linear polyacenes.

<table>
<thead>
<tr>
<th>No. of rings</th>
<th>NMR signals</th>
<th>Intensity Pattern</th>
</tr>
</thead>
<tbody>
<tr>
<td>2 (naphthalene)</td>
<td>3</td>
<td>1:2:2</td>
</tr>
<tr>
<td>3 (anthracene)</td>
<td>4</td>
<td>1:2:2:2</td>
</tr>
<tr>
<td>4</td>
<td>5</td>
<td>1:2:2:2:2</td>
</tr>
<tr>
<td>5</td>
<td>6</td>
<td>1:2:2:2:2:2</td>
</tr>
<tr>
<td>6</td>
<td>7</td>
<td>1:2:2:2:2:2:2</td>
</tr>
<tr>
<td>7</td>
<td>8</td>
<td>1:2:2:2:2:2:2:2</td>
</tr>
<tr>
<td>8</td>
<td>9</td>
<td>1:2:2:2:2:2:2:2:2</td>
</tr>
<tr>
<td>9</td>
<td>10</td>
<td>1:2:2:2:2:2:2:2:2</td>
</tr>
<tr>
<td>10</td>
<td>11</td>
<td>1:2:2:2:2:2:2:2:2:2</td>
</tr>
</tbody>
</table>
REFERENCES

47. K. Balasubramanian, manuscript in preparation.
49. K. Balasubramanian, submitted for publication.