Enumerating Perfect Matchings in a Ring of Diamonds

Shiena P. Tejada and Rhudaina Z. Mohammad
College of Science of Mathematics
Western Mindanao State University

Abstract

This paper presents an algorithm for enumerating all possible perfect matchings in a ring of diamonds – a connected claw-free cubic graph in which every vertex is in a diamond. A perfect matching in a graph refers to a collection of disjoint edges (a matching) in which every vertex of the graph is incident to exactly one edge of the matching, that is, it covers all vertices of the given graph. Our algorithm is based on Sang-il Oum's (2011) work on perfect matchings of claw-free cubic graphs. In addition, we give a proof by mathematical induction to show that there are exactly 2n + 1 perfect matchings in a ring of n diamonds – a result stated in Sang-il Oum's (2011) paper, but not explicitly proven. Finally, we present a pseudocode of our numerical scheme and some computational examples. In particular, we implement the algorithm using the Fortran programming language and plot the results using a command-line Gnuplot program.

Recommended citation based on APA 6th edition

R.Z.Mohammad (2016). Enumerating Perfect Matchings in a Ring of Diamonds. *Research Journal*, vol. 35, pp.40. www.wmsu.edu.ph/research_journal

Introduction

Graph theory is a field of mathematics that is continuously becoming more and more significant due to its varied applications in biochemistry (e.g., genomics, evolution), electrical engineering (e.g., communication networks, coding theory), computer science (e.g., algorithms, computations), and operations research (e.g., scheduling) (Pirzada & Dharwadker, 2007). Most of these real-world situations can be described by means of a diagram consisting a set of points together with lines joining certain pairs of these points (Bondy & Murty, 1982), which we call a "graph". More specifically, a graph G is an ordered triple (V(G),E(G),ψ_G) consisting of a nonempty set V(G) of vertices, a set E(G), disjoint from V(G), of edges, and an incidence function ψ_G that associates with each edge of G an unordered pair of (not necessarily distinct) vertices of G (Bondy & Murty, 1982). As an example, we can consider a graph in which every pair of distinct vertices are joined by an edge. We call such graph with n vertices, a complete graph K_n. A bipartite graph is one whose vertex set can be partitioned into two subsets V_1 and V_2, such that each edge has one end in V_1 and one end in V_2. Hence, a complete bipartite graph $K_{m,n}$ is bipartite graph in which every vertex in V_1 is joined to every vertex in V_2, with $|V_1|$=m and $|V_2|$= *n (Bondy & Murty, 1982)*. A graph is said to be cubic if every vertex on the graph has exactly three incident edges. Moreover, by "claw-free", we refer to a graph that has no induced subgraph isomorphic to the complete bipartite graph $K_{1,3,}$ which diagrammatically looks like a "claw" (Oum, 2011).

Most of the powerful combinatorial methods found in graph theory are related to a subfield of graph theory called matchings (Pirzada & Dharwadker, 2007). A subset M of the edge set E(G) is called a matching in G if no two elements are adjacent in graph G (Bondy & Murty, 1982). If every vertex of the graph is an end of exactly one edge in the matching, such matching is said to perfect. A well-known classical theorem of Petersen Petersen (1891) states that every cubic graph with no cutedge has a perfect matching. Sumner (1974) and Las Vergnas Las Vergnas (1974) independently showed that every connected claw-free graph with even number of vertices has a perfect matching. Both theorems imply that every claw-free cubic graph with no cutedge has at least one perfect matchings. Moreover, Lovasz and Plummer (2009) conjectured that every cubic graph with no cutedge has exponentially many perfect matchings (Petersen, 1891). In a recent paper, Sang-il Oum Sang-il Oum (2011) described the structure of 2-edge-connected claw-free cubic graph and classified into one of three: (1) isomorphic to complete graph K_4; (2) a ring of diamonds; or (3) built

from a 2-edge-connected cubic multigraph H by replacing some edges of H with strings of diamonds and replacing each vertex of H with a triangle. He also proved that every claw-free cubic n-vertex graph with no cut edge has more than $2^{n/12}$ perfect matchings.

Such perfect matchings can be applied in real life situations. As an example, consider a scheduling job problem on a machine (Chawla, 2007) where n jobs and m timeslots are available on a machine where these jobs can be performed. Each job has a set of valid slots when it can be performed, but no two jobs can be scheduled at the same time. The problem of finding the largest set of scheduled jobs translates to solving a bipartite matching problem (Chawla, 2007). Of course, all jobs must be performed.

In this case, a desired solution is one where each job is scheduled at exactly one valid timeslot. This becomes a perfect matching problem. Introducing more constraints (e.g. capacity of the machine, intermediate jobs) to the problem results in a complex perfect matching problem.

In this paper, we shall focus on enumerating all perfect matchings of a ring of diamonds – a connected claw-free cubic graph in which every vertex is in a diamond (an induced subgraph isomorphic to $K_4 \backslash e$ for some edge e of K_4. We start by counting the total number of perfect matchings in a ring of diamonds. Note that this result was stated in Oum (2011), but not explicitly proven. We provide the proof in this paper, employing the principle of mathematical induction. Further, we write an algorithm to enumerate all such perfect matchings. Finally, using the Fortran programming language (Markus, 2012), we present some numerical results.

Results and Discussion

In this section, we lay down a formal justification for the total number of perfect matchings of a ring graph of diamonds. Note that this result was simply stated, but not explicitly proven, in Sang-il Oum's paper entitled "Perfect Matchings in Claw-free Cubic Graphs" (Oum, 2011). We utilize a proof via the Principle of Mathematical Induction to establish the said result. Based on our proof, we then write

a pseudocode for enumerating all possible perfect matchings in a ring graph of any number of diamonds. Finally, we implement this algorithm using the Fortran programming language (Markus, 2012) and present the computational results via gnuplot plotting software (Williams & Kelley, 2015).

To begin, we provide the formal proof for the following theorem.

THEOREM. A ring of n diamonds has $2^n + 1$ perfect matchings.
Proof. We proceed by mathematical induction. We begin with the basis step and consider $n = 1$. Note that in this case, we have a ring graph with a single diamond. Observe that there are exactly 3 perfect matchings in this graph (see Figure 1). Two perfect matchings (independent edge set) can be found in the single diamond; while the other perfect matching involves the edge across the diamond. Further, we see that $3 = 2^1 + 1$, which proves the theorem for $n = 1$.

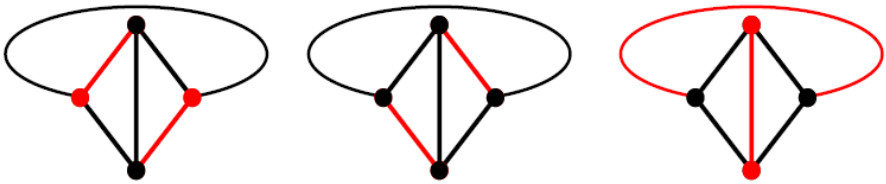

Figure 1. The 3 perfect matchings (in red) of a ring of a single diamond.

Now, for the inductive step, we assume that a ring of k diamonds ($k \in Z$) has a total number of $2^k + 1$ perfect matchings. We wish to prove that adding one more diamond graph in the ring resulting in a ring of $k + 1$ diamonds would have $2^k + 1 + 1$ perfect matchings.

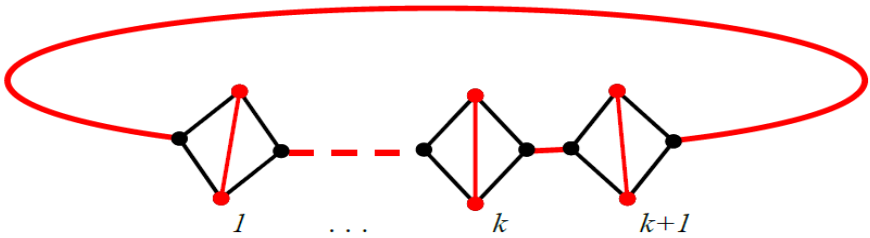

Figure 2. Adding one diamond graph in the ring constitutes exactly one across-diamond perfect matching (in red).

Note that adding one diamond graph to the ring does not change the number of perfect matchings across diamonds, that is, the perfect matchings involving the edges connecting the diamond graphs (see Figure 2). This only extends the independent edge set of the single across-diamond perfect matching. All there's left to do is count the number of perfect matchings not containing the edges connecting the diamond graphs. This means we simply count the perfect matchings within a diamond graph. By inductive hypothesis, we have 2^k within-diamond perfect matchings in a ring of k diamonds. Moreover, from our basis argument, we see that there are 2 within-diamond perfect matchings in a single diamond. Thus, adding one diamond in a ring of k diamonds implies that each of these 2^k within-diamond perfect matchings in a ring of *k* diamonds gives 2 within-diamond perfect matchings. Hence, by the basic counting principle, a ring graph with $k + 1$ diamonds would have a total of $2^k + 1$ within-diamond perfect matchings. Adding the single across-diamond perfect matching of the ring graph would then result in a total of $2^k + 1 + 1$ perfect matchings.

Thus, by the Principle of Mathematical Induction, a ring graph of n diamonds has a total of $2^n + 1$ perfect matchings. Q.E.D.

This theorem gives an insight on an algorithm for enumerating all possible perfect matchings of a ring of any given number of diamonds. We write its pseudocode as follows.

ALGORITHM. Enumerating Perfect Matchings of a Ring of Diamonds
Given a ring of *N* diamonds, we find all perfect matchings as follows:
input: number of diamonds *N*
output: perfect matchings in a ring $P(2^N + 1 , 6^N)$

```
1 initialize P = 0
2 while 1 ≤ i ≤ N do
3       while 1 ≤ j ≤ 4 do
4               set V_ij                        (define vertex of each diamond)
5       end do
6 end do
7 while 1 ≤ i ≤ N do
8       E(6(i – 1) + 1) = (V_i1 , V_i2)         (define edges within a diamond)
9       E(6(i – 1) + 2) = (V_i1 , V_i3)
10      E(6(i – 3) + 1) = (V_i2 , V_i3)
11      E(6(i – 1) + 4) = (V_i2 , V_i4)
12      E(6(i – 1) + 5) = (V_i3 , V_i4)
```

```
13      if i < N then
14      E(6(i – 1) + 6) = (V_{i1} , V_{i+1, 4})          (define edges across diamonds)
15      else
16              E(6(i – 1) + 6) = (V_{N1} , V_{14})
17      end if
18 end do
19 while 1 ≤ i ≤ N do
20      P(2^N + 1 , 6 (i – 1) + 6) = 1                   (mark first perfect matching)
21      P(2^N + 1 , 6 (i – 1) + 3) = 1
25 end do
25 while 1 ≤ j ≤ N do
26      while 1 ≤ k ≤ 2^j do
27              while 1 ≤ l ≤ 2^{N–j} do
28                      set i = 2 N – j (k – 1) + l
29                      if k ≡ 1 (mod 2)  then
30                              set C(i , j) = 1
31                      else
32                              set C(i , j) = 2
33                      end if
34              end do
35      end do
36 end do
39 while 1 ≤ i ≤ 2^N 1 do
40      while  1 ≤ j ≤ N do
41              if C(i , j) = 1 then
42      P (i , 6 ( j – 1) + 1) = 1          (mark within-diamond perfect matching)
43                      P (i , 6 ( j – 1) + 5) = 1
44              else
45                      P (i , 6 ( j – 1) + 2) = 1
46                      P (i , 6 ( j – 1) + 4) = 1
47              end if
48      end do
49 end do
50 stop
```

NUMERICAL RESULTS. We execute the above algorithm using the Fortran programing language (via Force application version 2.0.9p) [4], and visualize the resulting perfect matchings using gnuplot 4.6 (a portable command-line driven graphing utility for Linux, OS/2, MS Windows, OSX, VMS, and many other platforms) [10].

For our first experiment, we consider a ring graph with $N = 2$ diamonds.

By the Theorem, we see that there must be $2^2 + 1 = 5$ perfect matchings. To enumerate all 5 perfect matchings, we implement the Algorithm and obtain the computational results in Figure 3. Here, the first picture is the resulting single across-diamond perfect matching; and the succeeding four pictures are the within-diamond perfect matchings in a ring graph with 2 diamonds.

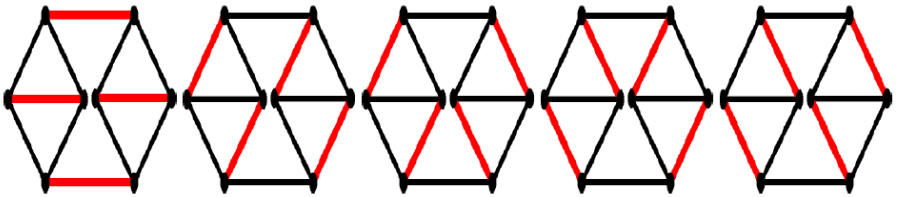

Figure 3. Five perfect matchings (in red) of a ring graph with 2 diamonds generated via our algorithm.

To better illustrate a ring of diamonds, we equally spread out the N diamonds in a circular ring. This involves subdividing the circle into N equal regions where we place the N diamond graphs.

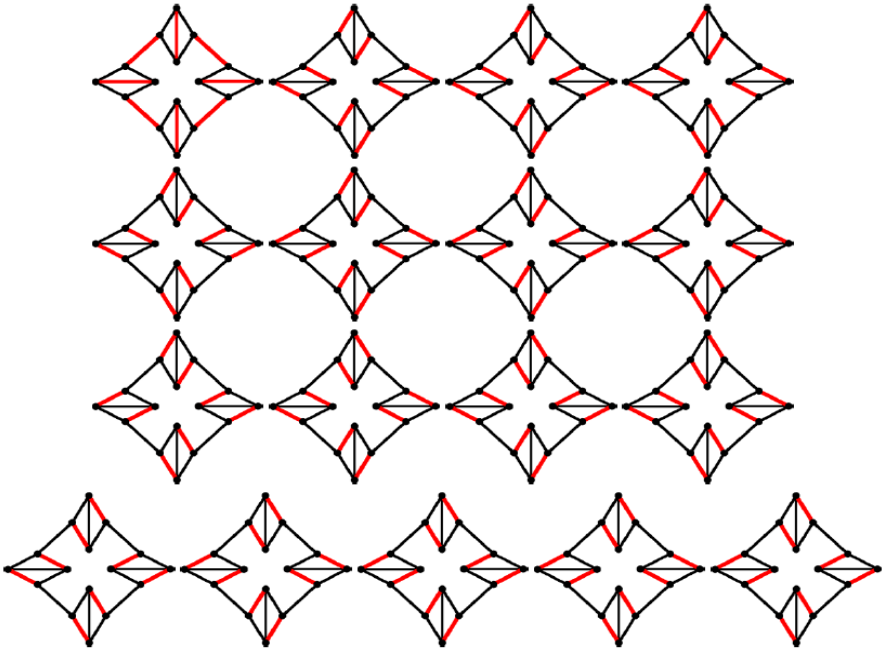

Figure 4. 17 perfect matchings (in red) of a 4-diamond ring generated via our algorithm.

This is evident in our second experiment (see Figure 4) where we enumerated all resulting $17 = 2^4 + 1$ perfect matchings for a ring graph with $N = 4$ diamonds. As in Figure 1, the first picture of Figure 2 denotes the across-diamond perfect matching and the succeeding pictures are those of the within-diamond perfect matchings.

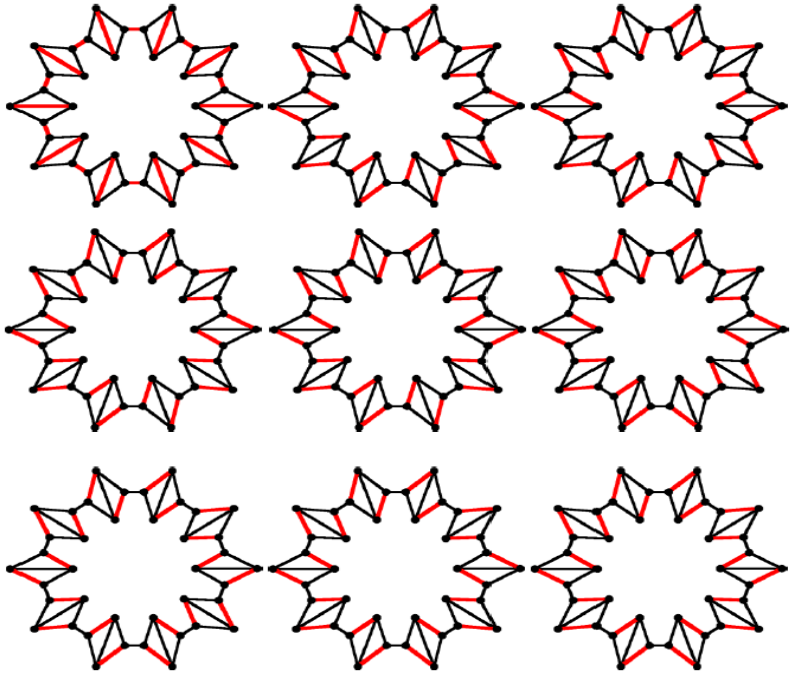

Figure 5. First 9 out of 1025 perfect matchings (in red) of a 10-diamond ring generated via Algorithm 1.

Finally, for a ring graph with $N = 10$ diamonds, we have $2^{10} + 1 = 1025$ perfect matchings. The first 9 of these 1025 resulting perfect matchings are shown in Figure 5. We emphasize that our algorithm can enumerate all 1025 perfect matchings, but for brevity, we only present 9 computational examples. It is however, important to note that as the number N of diamond graphs in a ring increases, computational time also increases. In particular, much of the computational cost is due to in finding all permutations resulting from the two types of within-single diamond perfect matching.

Summary and Conclusion

We have proven via the Principle of Mathematical Induction that a ring graph of n diamonds has a total of 2^n+1 perfect matchings. Based on this proof, we designed a scheme for enumerating all possible perfect matchings of a ring graph with any number n of diamond graphs. Our algorithm first produces the single across-diamond perfect matching, then generates all other 2^n within-diamond perfect matchings by marking appropriate edges starting from the first diamond to the last n diamond graph. We then executed this algorithm via the Fortran programming language and visualize the computational results using gnuplot plotting software. A technique used in illustrating the ring graphs of n diamonds was to subdivide the circle in equal parts for as to plot the ring graph in circular form. We found that as the higher the number of diamond graphs are in the ring, the more the algorithm is computationally expensive. Thus, we recommend that a better computational strategy be devised to enumerate all perfect matchings for large number N of diamond graphs in a ring. For further studies, we also wish to extend this work to other graphs, and also consider other types of matchings, in general.

References

Bondy, J.A. and Murty, U.S.R. (1982). Graph theory with applications. Elsevier Science Publishing Co., Inc.

Chawla, S (2007). Applications of network flow. A Lecture on Ad vanced Algorithms. University of Wisconsin. http://pages. cs.wisc.edu/~shuchi/courses/787-F09/scribe-notes/lec5.pdf

Las Vergnas, M. (1974). A note on matchings in graphs. Cahiers Centre Études Recherche Opér., 17(2-3-4), 257 – 260.

Lovasz, L., and Plummer, M. (2009). Matching Theory (vol. 367). Providence, USA: AMS Chelsea Publishing.

Markus, A. (2012). Modern Fortran in Practice. New York, YN, USA: Cambridge University Press.

Oum, S. (2011). Perfect matchings in claw-free cubic graphs. The Electronic Journal of Combinatorics, 18(1), 1-6.

Petersen, J. (1891). Die Theorie der regulären graphs. Acta Mathematica, 15(1), 193 – 220. Retrieved from http://projecteuclid.org/euclid.acta/1485881825. doi:10.1007/BF02392606.

Pirzada, S., and Dharwadker, A. (2007). Application of Graph Theory. Journal of the Korean Society for Industrial and Applied Mathematics 11(4), 19-38. Retrieved from http://mathnet.kaist.ac.kr/mathnet/thesis_file/03.pdf

Sumner, D. (1974). Graphs with 1-factors. Proceedings of the American Mathematical Society, 42, 8-12. https://doi.org/10.1090/S0002-9939-1974-0323648-6

Williams, T., and Kelley, C. (2015). Gnuplot 5.0 Reference Manual, Samurai Media Limited.