 doi:10.20944/preprints201911.0232.v1

# An Exact Parallel Algorithm for the Radio *k*-coloring Problem

E. M. Badr[1] and Khalid Aloufi[2]

[1]Scientific Computing Department, Faculty of Computers & Artificial Intelligence, Benha University, Benha, Egypt. badrgraph@gmail.com
[2]College of Computer Science & Engineering Taibah University Saudi Arabia
koufi@taibahu.edu.sa

**Abstract.**

For a positive integer $k$, a radio $k$-coloring of a simple connected graph $G = (V(G), E(G))$ is a mapping $|f(u)-f(v)| \geq k+1-d(u, v)$ such that $f: V(G) \to \{0,1,2,...\}$ for each pair of distinct vertices $u$ and $v$ of $G$, where $d(u, v)$ is the distance between $u$ and $v$ in $G$. The span of a radio $k$-coloring $f$, $rc_k(f)$, is the maximum integer it assigns to some vertex of $G$. The radio $k$-chromatic number, $rc_k(G)$ of $G$ is $\min\{rc_k(f)\}$, where the minimum is taken over all radio $k$-colorings $f$ of $G$. If $k$ is the diameter of $G$, then $rc_k(G)$ is known as the radio number of $G$. In this work, we propose four algorithms (two serial algorithms and their parallel versions) which related to the radio $k$-coloring problem. One of them is an approximate algorithm that determines an upper bound of the radio number of a given graph. The other is an exact algorithm which finds the radio number of a graph $G$. The approximate algorithm is a polynomial time algorithm while the exact algorithm is an exponential time algorithm. The parallel algorithms are parallelized using the Message Passing Interface (MPI) standard. The experimental results prove the ability of the proposed algorithms to achieve a speedup 7 for 8 processors.

**Keywords:** *Radio k*-coloring; radio number; MPI; parallel algorithm

## 1. Introduction

Motivated by the channel assignment problem proposed by Hale [1] in wireless networks, radio labeling in graphs has been studied from various perspectives. Let $G = (V, E)$ be a simple, connected and undirected graph. Let diam $(G) = d$ and

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 20 November 2019 doi:10.20944/preprints201911.0232.v1

$k \leq d$ be a positive integer. We use standard terms or notations as used in common texts such as [2, 3]. For a positive integer $k$, a radio $k$-coloring $f$ of a simple connected graph $G$ is an assignment of nonnegative integers to the vertices of $G$ such that for every two distinct vertices $u$ and $v$ of $G$, $|f(u)-f(v)| \geq k+1-d(u, v)$ The span of a radio $k$-coloring $f$, $rc_k(f)$, is the maximum integer assigned to some vertex of $G$ by $f$. The radio $k$-chromatic number, $rc_k(G)$ of $G$ is $\min\{rc_k(f)\}$, where the minimum is taken over all radio $k$-colorings $f$ of $G$. If $k$ is the diameter of $G$, then $rc_k(G)$ is known as the radio number and is denoted by $rn(G)$.

The radio $k$-coloring problem approaches to the radio coloring problem RCP when $k = 2$. RCP was first introduced by Griggs and Yeh [4] as the $L(2, 1)$ labelling problem. The RCP in general has been proved to be NP-complete [4], even when restricted to planar graphs, split graphs, or the complements of bipartite graphs [5, 6]. Exact results have been obtained for certain special class of graphs such as paths [7], cycles and trees [4].

Determining the radio number seems a difficult task, even for some basic graphs. For instance, the radio number for paths and cycles has been studied by Chartrand et al. [8, 9], in which, the bounds of the radio numbers for paths and cycles were presented. Later on, the radio numbers for paths and cycles were completely settled in [10].

Badr and Moussa [11] proposed an improved upper bound of the radio $k$-chromatic number for a given graph against another which is based on work by Saha and Panigrahi [12]. They introduced also a polynomial algorithm which (differs from the first algorithm and derived from Liu and Zhu [10]) which determines the radio number of the path graph $P_n$. They [11] also proposed a new integer linear programming model [13-16] for the radio $k$-coloring problem.

In this work, we propose four algorithms (two serial algorithms and their parallel versions) which related to the radio $k$-coloring problem. One of them is an approximate algorithm that determines an upper bound of the radio number of a given graph. The other is an exact algorithm which finds the radio number of a graph $G$. The approximate algorithm is a polynomial time algorithm while the exact algorithm is an exponential time algorithm. The parallel algorithms are parallelized using the Message Passing Interface (MPI) standard. The experimental results, analysis prove the ability of the proposed algorithms to achieve a speedup 7 for 8 processors.

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 20 November 2019 doi:10.20944/preprints201911.0232.v1

## 2. Main Contributions

In this section, we propose four algorithms (two serial algorithms and their parallel versions) related to the radio $k$-coloring problem. One of them is an approximate algorithm that determines an upper bound of the radio number of a given graph. The other is an exact algorithm which finds the radio number of a graph $G$. The approximate algorithm is a polynomial time algorithm while the exact algorithm is an exponential time algorithm.

### 2.1. An upper bound of the radio number for a given graph *G*.

Badr and Moussa [11] proposed an upper bound algorithm of the radio number for a given graph as Algorithm 1.

**Algorithm 1.** Finding a radio $k$-coloring of a graph

**Input**: $G$ be an $n$-vertex simple connected graph and $k$ be a positive integer.
the adjacency matrix $A[n][n]$ of $G$
**Output**: A radio $k$-coloring of $G$.

```
Begin
    Compute the distance matrix D[n][n] of G using Floyed-Warshall's algorithm and the
    adjacency matrix A[n][n] of G.
    Upper  = ∞;
    for  l = 1 to  n  do
          for i= 1 to n do
                labeling [i] = 0;
          end
          for i= 1 to n do
                      for j=1 to n do
                          c[i][j]=diam+1-D[i][j];
                      end
                      c[i][i] = ∞;
          end
          for i = 2  to  n  do
 /*find the minimum value m of the column with position  p*/
                     [m, p] = min [c( l , : ) ];
                     for j = 1 to n
                               c[p][j] = c[p][j] + m
                               if c[p][j] < c[l][j]
                                    c[p][j] = c[l][j]
                               end
                    end
                    labeling [p] = m
                    l = p
          end
             /* find the max  value of  the labeling */
               Max_Value = max (labeling))
               if  Upper  >  Max_Value
                     Upper  = Max_Value
               end
    end
End
```

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 20 November 2019 doi:10.20944/preprints201911.0232.v1

### 2.1.1. Complexity of Algorithm 1:

Here we analyze the complexity of Algorithm 1. Algorithm 1 consists of two steps, the first step is Floyed-Warshall's algorithm and the other is the remiander of Algorithm 1. We know the complexity of Floyed-Warshall's algorithm is $O(n^3)$ because it has three inner for loops. The remainder of Algorithm 1 has three inner for loop only each of them has length $n$ so the complexity of step 2 is $O(n^3)$ . Thus the overall complexity for Algorithm 1 is

$$\approx 2O(n^2)+O(n^3)+O(n^3) \approx 2O(n^3) \approx O(n^3).$$

## 2.2. An exact sequential algorithm for determining the radio number of a given graph *G*.

In this section, we propose an exact algorithm (Algorithm 2) which determines the radio number of a given graph with n vertices. Recall the radio $k$-coloring problem is NP-complete problem so Algorithm 2 is an exponential time algorithm. It based on the brute force technique. Algorithm 2 determines the upper bound $k$ of the radio number of a graph $G$ with order $n$ then it generates all possible labeling sets of the graph $G$. It is clear that the number of all labeling sets for the graph $G$ is $k\ (k-1)\,(k-2)\cdots(k-n+1)$. Not of all these labeling sets satisfy the following inequality: $|f(u)\text{-}f(v)| \geq k+1\text{-}d(u, v)$ so the following step in Algorithm 2 checks whether or not a certain set whether satisfies the above inequality. Finally, Algorithm 2 determines the radio number of the graph G by taking $\{\min(\max(L_i)) : 1 \leq i \leq k\}$ where $L_i$ are the all possible labeling sets for $G$.

**Algorithm** 2: A serial algorithm for finding the radio number of a given graph

**Input**: $G$ be an $n$-vertex simple connected graph and $k$ be a positive integer.
the adjacency matrix $A[n][n]$ of $G$

**Output**: A radio $k$-coloring of $G$.

**Begin**

Call Algorithm 1 (Finding a radio $k$-coloring of a graph)

```
c = 1;                          % number of all labeling sets
for i = 1: n
    c = c *(upper - i+1);
end
E(1:c, 1: n)=0;
i = 1;
while i <= c
    j=0;
    while j <n
        j=j+1;
        num = (rand()*upper)+1;
       num=fix(num);
        Lia = ismember(num,E(i,:));
```

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 20 November 2019 doi:10.20944/preprints201911.0232.v1

```
            if Lia==1
                  j = j-1;
                  continue;
            end
            E(i,j)=num;
      end
      Lia=0;
     if  (i >1)
          for l=i:-1:2
               Lia=isequal(E(i,:),E(l-1,:));
               Lia=Lia +0;
          end
       end
       if Lia ==1
            E(i,:)=0;
            continue;
       end
       i =i+1;
  end
% ********************check the radio number*********
  Radio = ∞;
  for i=1:c
      for j=1:n
        for k=j+1:n
            if    abs ( E(i,j)- E(i,k) ) >= diam+1-D(j,k)
                  if Radio > max(E(i,: ) )
                        Radio = max(E(i,: ) );
                        index = i;
                  end
            end
        end
      end
   end
End /* Begin
```

### 2.2.1. Complexity of Algorithm 2:

In this section, we analyze the complexity of Algorithm 2. Algorithm 2 consists of more than one procedure: the first procedure is Floyed-Warshall's method. We know the complexity of Floyed-Warshall's algorithm is $O(n^3)$ because it has only three inner for loops. The second procedure of Algorithm 2 has three inner for loop only each of them has length $n$ so the complexity of the second procedure is $O(n^3)$. The remain of Algorithm 1 has the checking subroutine which has complexity $O(k^n)$ because the first vertex in $G$ takes $k$ possibilities, the second vertex takes ($k$-1) possibilities, and so on until the last vertex takes ($k$-$n$+1) possibilities. Thus the overall complexity for the Algorithm 1 is

$$\approx O(n^3)+O(n^3)+O(k^n)\approx 2\,O(n^3)+O(k^n)\approx O(k^n).$$

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 20 November 2019 doi:10.20944/preprints201911.0232.v1

### 2.3. A Parallel Version of Algorithm 1 (for finding the upper bound of the radio number for a given graph *G*

In this section, we propose a parallel version of Algorithm 1, namely PAlgorithm 1, which determines the upper bound of the radio number of a given graph *G*. In PAlgorithm 1, each processor assigns the label zero to $\frac{n}{p}$ number of vertices where *n* and *p* are the order of *G* and the number of processors respectively. It is clear that balance loading occurs if mod (*n*, *p*) = 0, otherwise unbalance loading occurs.

**Algorithm 3. Parallel Version of Algorithm 1:** Finding an upper bound of the radio *k*-coloring of a graph *G*

**Input**: Let G be an n-vertex, simple, connected graph, k be a positive integer and A[n][n] be the adjacency matrix of *G*

**Output**: An upper bound of the radio *k*-coloring of *G*

**Begin**

1- For all processors

```
for (j = rank+1; j = n ; j = j +p)
    l = j            % l is the index of he first loop in Algorithm 1
    Call Algorithm 1
end
```

2- Gather the Local_ Upper number from each processors

for $0 \le i < p$ pardo

Send *Local_* Upper to processor 0

In processor 0

Search *rank* which corresponding to *min*(Upper)

*min_rank* = *rank*

end

3- From processor *min_rank*

Print *Local_* Upper % as the upper bound of the radio number of the graph *G*

**End**

### 2.4. A Parallel Version of Algorithm 2

In this section, we propose a parallel version of Algorithm 2, namely PAlgorithm 2, which determines the radio number of a given graph *G*. The main idea of PAlgorithm 2 is that every vertex of the graph *G* is labeled $\frac{c}{k}$ times by the integers 1, 2, 3, . . . , *k* where *c* is all possible combinations of *n* vertices by the upper bound *k* of the radio number of *G* which is determines by Procedure 2. The following example explains the above main idea of PAlgorithm 2.

**Example 1:** Consider a graph *G* with order *n* = 3 and upper bound *k* = 4 of the radio number of *G*. Now, we determine the all possible combinations [ c = *k* (*k*-1) (*k*-2) (*k*-

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 20 November 2019 doi:10.20944/preprints201911.0232.v1

3) … ($k$-$n$+1) = 4 * 3 * 2 = 24] of $n$ vertices by the upper bound $k$ of the radio number of $G$ as shown:

**Table 1: All possible combinations for the graph $G$ with order $n$ and upper bound $k$ = 4**

| $v_1$ | $v_2$ | $v_3$ | $v_1$ | $v_2$ | $v_3$ | $v_1$ | $v_2$ | $v_3$ | $v_1$ | $v_2$ | $v_3$ |
|---|---|---|---|---|---|---|---|---|---|---|---|
| 1 | 2 | 3 | 2 | 1 | 3 | 3 | 1 | 2 | 4 | 1 | 2 |
| 1 | 2 | 4 | 2 | 1 | 4 | 3 | 1 | 4 | 4 | 1 | 3 |
| 1 | 3 | 2 | 2 | 3 | 1 | 3 | 2 | 1 | 4 | 2 | 1 |
| 1 | 3 | 4 | 2 | 3 | 4 | 3 | 2 | 4 | 4 | 2 | 3 |
| 1 | 4 | 2 | 2 | 4 | 1 | 3 | 4 | 1 | 4 | 3 | 1 |
| 1 | 4 | 3 | 2 | 4 | 3 | 3 | 4 | 2 | 4 | 3 | 2 |

From above example, it is clear that the first vertex is labeled $\frac{c}{k} = \frac{24}{4} = 6$ times by integer 1. Now, we can broadcast the data on all processors according to the label of the first vertex as follows:

```
for ( j = rank + 1; j = k; j = j + p)
      if E(i,1) == j
            for ( i = 1; c; i++)
                  Algorithm 1
            end
      end
end
```

**Algorithm 4. Parallel Version of Algorithm 2** Finding a radio $k$-coloring of graph $G$

**Input**: Let G be an n-vertex, simple, connected graph, k be a positive integer, and A[n][n] be the adjacency matrix of G

**Output**: A radio k-coloring of G.

**Begin**

1- Call PAlgorithm 1 % finding an upper bound of the radio number of G by parallel computing

2- For all processors

```
 for ( j = rank + 1; j = k; j = j + p)
       if E(i,1) == j
           for ( i = 1; c; i++)
                 Algorithm 2
           end
       end
  end
```

3- Gather the Local_ Radio number from each processors

for $0 \le i < p$ pardo

Send *Local_Radio* to processor 0

In processor 0

Search *rank* which corresponding to *min*(*Radio*)

*min_rank* = *rank*

end

4- From processor *min_rank*

Print *Local_Radio* % as the radio number of the graph $G$

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 20 November 2019 doi:10.20944/preprints201911.0232.v1

**End**

## 3. Computational Study

The triangular snake graph $\Delta_k$ is obtained from the path $P_n$ by replacing each edge of the path by a triangle $C_3$ [17] where $k$ is the number of blocks. A connected graph in which the $k$ blocks are isomorphic to the cycle $C_n$ and the block-cutpoint graph is a path denoted by $kC_n$-*snake* [18]. Here we describe the numerical experiments then present computational results, which demonstrate the efficiency of the proposed algorithms on a set of test problems: paths, cycles, $\Delta_k$ -*snakes,* $kC_4$-*snakes* and $kC_6$-*snakes* . All test runs were carried out on a computer cluster of the Faculty of Science [19], Cairo University, a homogeneous PC cluster consists of 9 nodes, one master and eight slaves. Each slave node has an Intel(R) core (TM)2 Duo CPU E7400 2.80 GHz and 4 Gb DDR2 RAM and master node has a microprocessor Intel(R) core(TM)2 Quad CPU Q6700 2.66 GHz microprocessor and 4 Gb DDR2 RAM. The processors were interconnected using Fast Ethernet and Scalable Coherent Interface (SCI). By using the message passing interface library MPI and C language, we can write a code for parallel computing simply using the single program multiple data (SPMD) approach.

The abbreviations Ub0, Ub1 and Ub2 are used to denote upper bounds are due to Badr and Moussa [11], Saha and Panigrahi [12] and Badr and Moussa [11] respectively. In [11], we emphasize that Ub0 is based on the integer liner programming model but Ub2 is based on the polynomial time algorithm which its complexity is $O(n^3)$. On the other hand, St and Ex are used to denote the exact radio numbers are due to Liu and Zhu [10] and the proposed Algorithm 2 respectively.

**Table 2: Comparison among Ub0, Ub1, Ub2, St and Ex for the radio number of path graph $rn(P_n)$**

| *N* | *Ub0* | *Ub1* | *Ub2* | St | Ex |
|---|---|---|---|---|---|
| 1 | 0 | 0 | 0 | 0 | 0 |
| 2 | 1 | 1 | 1 | 1 | 1 |
| 3 | 4 | 3 | 3 | 3 | 3 |
| 4 | 9 | 6 | 5 | 5 | 5 |
| 5 | 16 | 11 | 10 | 10 | 10 |
| 6 | 25 | 15 | 14 | 13 | 13 |
| 7 | 36 | 22 | 20 | 20 | 20 |
| 8 | 49 | 28 | 26 | 25 | 25 |
| 9 | 64 | 37 | 35 | 34 | 34 |
| 10 | 81 | 45 | 42 | 41 | 41 |
| 11 | 100 | 56 | 53 | 52 | - |
| 12 | 121 | 66 | 62 | 61 | - |
| 13 | 144 | 80 | 75 | 74 | - |
| 14 | 169 | 91 | 86 | 85 | - |

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 20 November 2019 doi:10.20944/preprints201911.0232.v1

| | | | | | |
|---|---|---|---|---|---|
| 15 | 196 | 107 | 101 | 100 | - |
| 16 | 225 | 120 | 114 | 113 | - |
| 17 | 256 | 138 | 132 | 130 | - |
| 18 | 289 | 153 | 146 | 145 | - |
| 19 | 324 | 173 | 166 | 164 | - |
| 20 | 361 | 190 | 182 | 181 | - |
| 21 | 400 | 213 | 204 | 202 | - |
| 22 | 441 | 231 | 222 | 221 | - |
| 23 | 485 | 256 | 246 | 244 | - |
| 24 | 525 | 276 | 266 | 265 | - |
| 25 | 580 | 303 | 293 | 290 | - |

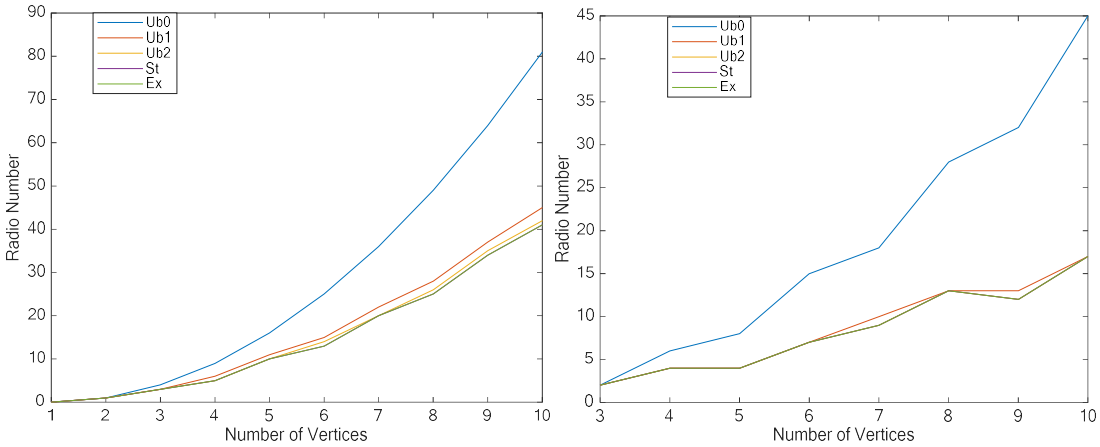



**Figure1: Comparison among Ub0, Ub1, Ub2 and St, Ex for a) Path Graph and b) Cycle Graph**

Table 2, Table 3 and Figure 1 show Algorithm 1 (Ub2) overcomes both the algorithm which based on integer linear programming [11] Ub0 and the algorithm Ub1 was proposed by Saha and Panigrahi [12].We also note that Algorithm 2 (Ex) outperforms other algorithms but the proposed Algorithm 2 (Ex) and the algorithm (St) was proposed by Liu and Zhu [10] are equivalent. As well the proposed Algorithm 2 (Ex) is better than the Algorithm (St) because the proposed Algorithm 2 (Ex) determines the exact radio number for a given graph but the Algorithm (St) finds the exact radio numbers for special graphs (path graph and cycle graph) only.

**Table 3: Comparison among Ub0,Ub1, Ub2, St and Ex for the radio number of cycle graph**

$rn(C_n)$

| ***N*** | ***Ub0*** | ***Ub1*** | ***Ub2*** | **St** | **Ex** |
|---|---|---|---|---|---|
| 1 | - | - | - | - | - |
| 2 | - | - | - | - | - |
| 3 | 2 | 2 | 2 | 2 | 2 |
| 4 | 6 | 4 | 4 | 4 | 4 |
| 5 | 8 | 4 | 4 | 4 | 4 |
| 6 | 15 | 7 | 7 | 7 | 7 |
| 7 | 18 | 10 | 9 | 9 | 9 |
| 8 | 28 | 13 | 13 | 13 | 13 |
| 9 | 32 | 13 | 12 | 12 | 12 |
| 10 | 45 | 17 | 17 | 17 | 17 |

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 20 November 2019 doi:10.20944/preprints201911.0232.v1

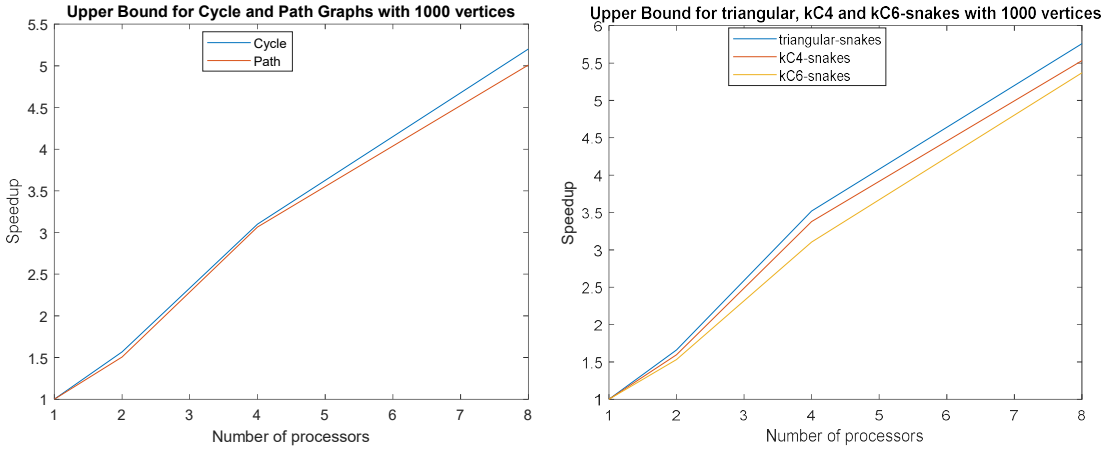
| | | | | | |
|---|---|---|---|---|---|
| 11 | 50 | 21 | 20 | 20 | - |
| 12 | 66 | 26 | 26 | 26 | - |
| 13 | 72 | 25 | 24 | 24 | - |
| 14 | 91 | 31 | 31 | 31 | - |
| 15 | 98 | 36 | 35 | 35 | - |
| 16 | 120 | 43 | 43 | 36 | - |
| 17 | 128 | 44 | 40 | 32 | - |
| 18 | 153 | 49 | 49 | 41 | - |
| 19 | 162 | 55 | 54 | 36 | - |
| 20 | 190 | 64 | 64 | 46 | - |
| 21 | 200 | 64 | 60 | 40 | - |
| 22 | 231 | 71 | 71 | 51 | - |
| 23 | 253 | 78 | 77 | 44 | - |
| 24 | 276 | 89 | 89 | 56 | - |
| 25 | 300 | 89 | 84 | 48 | - |

In Table 2 and Table 3, it is clear that the proposed Algorithm 2 (Ex) determine the radio number for paths and cycle with the maximum number of vertices 11 only because of the Algorithm 2 is exponential time algorithm with complexity $O(k^n)$. In order to overcome this obstacle, we use more processors for testing graphs which have more than 11 vertices.

**Table 4: PAlgorithm 1 ( the upper bound of the radio number) for size 1000 of cycle graph $C_n$ and path graph $P_n$**

| Size / Proc. | 1 | 2 | 4 | 8 |
|---|---|---|---|---|
| $C_{1000}$ | 146.0418 | 93.1508 | 47.1087 | 28.0769 |
| Speed up | 1.00000 | 1.5678 | 3.1001 | 5.2015 |
| $P_{1000}$ | 113.8448 | 75.6695 | 37.1350 | 22.7231 |
| Speed up | 1.0000 | 1.5045 | 3.0657 | 5.0101 |

From the superiority of Algorithm 1, we have a motivation to introduce the parallel version of algorithm 1, namely PAlgoritm 1.




(a) (b)

**Figure 2: Determining the upper bound of the radio number on (a) Cycle and path graphs with 1000 vertices and (b) $\Delta_k$ *-snakes*, $kC_4$ *-snakes* and $kC_6$ *-snakes* with about 1000 vertices on eight processors**

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 20 November 2019 doi:10.20944/preprints201911.0232.v1

Table 4 and Table 5 have the parallel results for determining the upper bound of the radio number of cycle and path graphs with 1000 vertices and $\Delta_k$ *-snakes* , $kC_4$ *-snakes* and $kC_6$ *-snakes* with about 1000 vertices on eight processors. From these tables, we can observe that speedup of PAlgorithm 1 is about 6. We also note that the maximum number of vertices is 1000 vertices because Algorithm 1 is polynomial time algorithm but Algorithm 2 is exponential time algorithm with complexity $O(k^n)$ so it run on 11 vertices only.

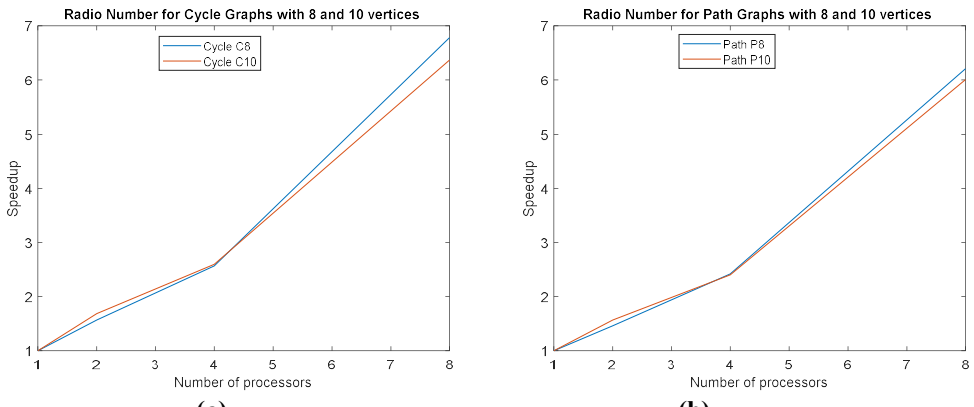
**Table 5: PAlgorithm 1 (the upper bound of the radio number) of $\Delta_{500}$ *-snakes* , $333C_4$ *-snakes* and $200C_6$ *-snakes***

| Size / Proc. | 1 | 2 | 4 | 8 |
|---|---|---|---|---|
| $\Delta_{500}$ ***-snakes*** | 124.2723 | 74.8313 | 35.2906 | 21.5747 |
| Speed up | 1.0000 | 1.6607 | 3.5214 | 5.7601 |
| $333C_4$ ***-snakes*** | 142.2393 | 88.9886 | 42.0814 | 25.6986 |
| Speed up | 1.0000 | 1.5984 | 3.3801 | 5.5349 |
| $200C_6$ ***-snakes*** | 34.2044 | 22.3252 | 11.0173 | 6.3678 |
| Speed up | 1.0000 | 1.5321 | 3.1046 | 5.3715 |

From the superiority of Algorithm 2, we have a motivation to introduce the parallel version of Algorithm 2, namely PAlgoritm 2.

**Table 6: PAlgorithm 2 (the radio number) for (8 and 10) vertices of cycle graph $C_n$ and path graph $P_n$**

| Size / Proc. | 1 | 2 | 4 | 8 |
|---|---|---|---|---|
| $C_8$ | 447.3240 | 285.4652 | 174.3681 | 65.9682 |
| Speed up | 1.0000 | 1.5670 | 2.5654 | 6.7809 |
| $C_{10}$ | 1936.8719 | 1145.4943 | 745.3234 | 304.0185 |
| Speed up | 1.0000 | 1.6901 | 2.5987 | 6.3709 |
| $P_8$ | 429.7530 | 293.7679 | 177.3567 | 69.1978 |
| Speed up | 1.0000 | 1.4629 | 2.4231 | 6.2105 |
| $P_{10}$ | 1936.8719 | 1232.8911 | 806.6266 | 322.2695 |
| Speed up | 1.0000 | 1.5710 | 2.4012 | 6.0101 |




**(a)** **(b)**

**Figure 3: Determining the radio number of (a) Cycle and (b) Path graphs with 8 vertices and 10 vertices on 8 processors**

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 20 November 2019 doi:10.20944/preprints201911.0232.v1

Table 6 and Table 7 have the parallel results for determining the radio number of cycle and path graphs with 8 and 10 vertices and $\Delta_k$ *-snakes* , $kC_4$ *-snakes* and $kC_6$ *-snakes* with about 10 vertices on eight processors. It is clear that the number of vertices is very small because the Algorithm 2 is an exact exponential time algorithm with complexity $k^n$. We can use more processors for testing more vertices. From these tables, we can observe that the speedup of PAlgorithm 2 is about 7.

**Table 7: PAlgorithm 2 (the radio number) of $\Delta_4$ *-snakes* , $3C_4$ *-snakes* and $2C_6$ *-snakes***

| Size / Proc. | 1 | 2 | 4 | 8 |
|---|---|---|---|---|
| $\Delta_4$ ***-snakes*** | 1765.8947 | 1205.0598 | 479.9801 | 259.9963 |
| Speed up | 1.0000 | 1.4654 | 3.6791 | 6.7920 |
| $3C_4$ ***-snakes*** | 1975.9780 | 1259.6277 | 559.4502 | 328.0503 |
| Speed up | 1.0000 | 1.5687 | 3.5320 | 6.0234 |
| $2C_6$ ***-snakes*** | 2374.3218 | 1572.7110 | 597.7949 | 360.1932 |
| Speed up | 1.0000 | 1.5097 | 3.9718 | 6.5918 |

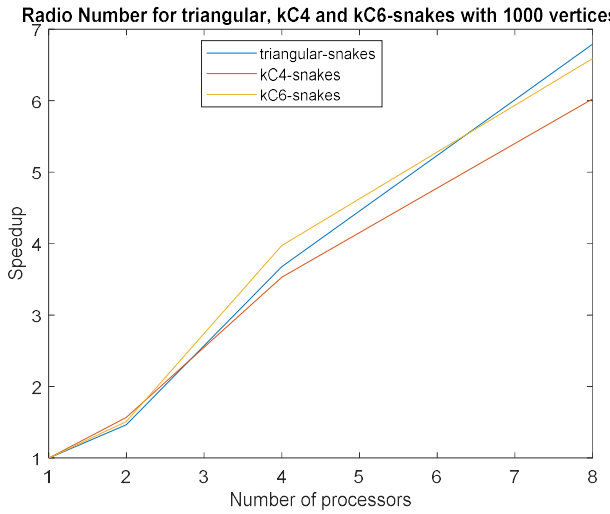



**Figure 4: Determining the radio number of $\Delta_k$ *-snakes*, $kC_4$ *-snakes* and $kC_6$ *-snakes* with about 10 vertices on eight processors**

## 4. Conclusion

In this work, we proposed four algorithms (two serial algorithms and their parallel versions) which related to the *radio k-coloring* problem. One of them is an approximate algorithm that determines an upper bound of the radio number for a given graph. The other is an exact algorithm which finds the radio number of a graph *G*. The approximate algorithm is a polynomial time algorithm while the exact algorithm is an exponential time algorithm. The parallel algorithms were parallelized using the Message Passing Interface (MPI) standard. The experimental results,

 doi:10.20944/preprints201911.0232.v1

analysis proved the ability of the proposed algorithms to achieve a speedup 7 for 8 processors.

## 5. Reference

[1] W. K. Hale, Frequency assignment: Theory and applications, Proceedings of the IEEE 68 (12) (1980), 1497–1514.

[2] R. Balakrishnan, K. Renganathan, A Textbook of Graph Theory, Springer (2000).

[3] G. Chartrand, P. Zhang, Introduction to Graph Theory. McGraw-Hill (2006).

[4] J. R. Griggs and R. K. Yeh. Labeling graphs with a condition at distance 2. SIAM Journal on Discrete Mathematics, 5:586-595, 1992.

[5] *Broersma, Hajo (2005),* "A general framework for coloring problems: old results, new results, and open problems"*, Combinatorial geometry and graph theory, Lecture Notes in Comput. Sci.,* ***3330****, Springer, Berlin, pp. 65–79*

[6] Bodlaender, Hans L.*; Kloks, Ton; Tan, Richard B.; van Leeuwen, Jan (2000), "λ-coloring of graphs", STACS 2000: 17th Annual Symposium on Theoretical Aspects of Computer Science, Lille, France, February 17–19, 2000, Proceedings, Lecture Notes in Computer Science,* ***1770****, Springer, Berlin, pp. 395–406.*

[7] R. Y. Yeh. Labeling Graphs with a Condition at Distance Two. PhD thesis, Deptartment of Mathematics, University of South Carolina, Columbia, SC, 1990

[8] G. Chartrand, D. Erwin, F. Harary and P. Zhang, Radio labelings of graphs, Bull. Inst. Combin. Appl., 33 (2001), 77-85.

[9] G. Chartrand, D. Erwin, and P. Zhang, A graph labeling problem suggested by FM channel restrictions, manuscript.

[10] D. Liu and X. Zhu, Multi-level distance labelings for paths and cycles, SIAM J. Discrete Math., 19(3), 610-621, (2005)

[11] Elsayed M. Badr, Mahmoud I. Moussa in *Wireless Networks (2019)*, An upper bound of radio *k*-coloring problem and its integer linear programming model, First Online: 18 March 2019.

[12] Saha L.,Panigrahi P. (2012) A Graph Radio *k*-Coloring Algorithm. In: Arumugan S., Smyth W. F. (eds) Combinatorial Algorithms. IWOCA 2012. Lecure Notes in Computer Science, vol 7643. Springer, Berlin, Heidelberg.

[13] M.S. Bazaraa, J.J. Jarvis, H.D. Sherali, Linear Programming and Network Flows, third ed., John Wiley, NY, 2004.

[14] G. Chartrand, L. Eroh, M.A. Johnson, and O.R. Oellermann: Resolvability in

graphs and the metric dimension of a graph, *Discrete Appl. Math.*,105 (2000), 99-113.

[15] E. S. Badr, K. Paparrizos, N. Samaras, and A. Sifaleras (2005), On the Basis Inverse of the Exterior Point Simplex Algorithm, in Proc. of the 17th National Conference of Hellenic Operational Research Society (HELORS), 16-18 June, Rio, Greece, pp. 677-687.

[16] E.S. Badr, K. Paparrizos, Baloukas Thanasis and G. Varkas (2006), Some computational results on the efficiency of an exterior point algorithm, in Proc. of the 18th National Conference of Hellenic Operational Research Society (HELORS), 15-17 June, Rio, Greece, pp. 1103-1115

[17] R. Ponraj and S. Sathish Narayanan (2015) " Mean cordiality of some snake graphs, Palestine Journal of Mathematics" Vol. 4(2), 439–445.

[18] E. M. Badr (2015) " On Graceful Labeling of the Generalization of Cyclic Snakes" Journal of Discrete Mathematical Sciences and Cryptography 18 (6), 773-783, 2015.

[19] Sweilam NH, Moharram HM, Sameh Ahmed. On the parallel iterative finite difference algorithm for 2-D Poisson's equation with MPI cluster. In: The 8th international conference on INFOrmatics and systems (INFOS2012), IEEE Explorer; 2012.

[20] E.S. Badr, M. Moussa, K. Paparrizos, N. Samaras, and A. Sifaleras, Some computational results on MPI parallel implementation of dense simplex method, World Academy of Science, Engineering and Technology (WASET), 23, 2008, 778–781.