On Node Ranking of Graphs

Yung-Ling Lai, Sheng-Hsiung Wang and Wei-Chun Lin
Department of Computer Science and Information Engineering
National Chiayi University, Chiayi, Taiwan
{yllai, s0942783, s0942810}@mail.ncyu.edu.tw

Abstract

For a connected graph $G=(V,E)$, a node ranking labeling $C : V \to \{1,2,...,k\}$ is a mapping such that for every path between any two vertices $u,v$ with $C(u)=C(v)$, there exists at least one vertex $w$ on the path with $C(w)>C(u)=C(v)$. The value $C(v)$ is called the rank of the vertex $v$ and $G$ is said to be $k$-rankable if there exists a node ranking labeling of $G$ with maximum rank $k$. The node ranking problem is the problem of finding minimum $k$ such that $G$ is $k$-rankable. There are two versions of node ranking problem, namely offline and online. In off-line version, the node ranking problem deals with a graph where all vertices and edges are known. In on-line version, the vertices are given one by one in an arbitrary order and only the edges of the induced subgraph of given vertices are known when the rank of each vertex has to be chosen. The rank can not be changed after it is assigned. This paper solved node ranking problem of corona graphs and mesh in off-line version, and gave on-line node ranking algorithm for sun graphs.

1 Introduction

Let $G=(V,E)$ be an undirected graph where $V$ denotes the set of vertices and $E$ denotes the set of edges. $G$ is said to be $k$-rankable if there is a mapping $C : V \to \{1,2,...,k\}$ such that every path between any two vertices $u,v$ with $C(u)=C(v)$, there exists at least one vertex $w$ on the path with $C(w)>C(u)=C(v)$. The mapping $C$ is called a node ranking labeling of $G$ and the value $C(v)$ is called the rank of the vertex $v$. The node ranking number $\chi(G)$ is the smallest integer $k$ such that $G$ is $k$-rankable. A node ranking labeling is an optimal node ranking labeling of $G$ if its maximum rank is $\chi(G)$. The node ranking problem is the problem of finding the node ranking number $\chi(G)$ of a graph $G$. We said the node ranking problem is solved if a formula for the node ranking number is obtained or an algorithm of an optimal node ranking labeling is given. The node ranking problem is applicable in communication network design[3][15]; finding the minimum height of a node separator tree of a graph [3][8][14], which are extensively used in VLSI layout [11][15]; and developing algorithms for planning efficient assembly of products in manufacturing systems [4] etc. There are two versions of node ranking problem, off-line and on-line versions. In off-line version, the graph with all vertices and edges are given in advance. In on-line version, the vertices are given one by one in an arbitrary order together with the edges adjacent to the vertices that are already given. The rank has to be assigned in real time and once the rank is assigned to a vertex, it is not changeable. Similar to the off-line version, the on-line node ranking number is denoted as $\chi^*_r(G)$, which is the smallest integer $k$ such that $G$ is on-line $k$-rankable.

Node ranking problem has been studied since 1980s. In the offline version, it is known that for a path $P_n$, $n \geq 1$, $\chi_r(P_n) = \lceil \log_2 n \rceil + 1$ [5]. For a cycle $C_n$, $n \geq 3$, $\chi_r(C_n) = \lceil \log_2 n \rceil + 1$ [1]. For wheel graphs $W_n$, $n \geq 3$, $\chi_r(W_n) = \lceil \log_2 n \rceil + 2$ [7]; the node ranking number of the complete bipartite graph $K_{n,n}$ is $\chi_r(K_{n,n}) = m+1$ for $m \leq n$ [8]; and star graphs $S_n = K_{1,n}$ with $n$ vertices, $\chi_r(S_n) = 2$ for $n \geq 3$ [13]. An upper bound of the node ranking number for an arbitrary tree with $n$ nodes was given by [3], they proposed an $O(n \log n)$ time optimal node ranking algorithm, which was further improved to $O(n)$ by [12]. There are fewer results in the on-line version. [1] gave tight bounds for paths and cycles as $\lceil \log_2 n \rceil + 1 < \chi^*_r(P_n) < 2 \lceil \log_2 n \rceil + 1$ for $n \geq 1$; and $\lceil \log_2 n \rceil + 1 < \chi^*_r(C_n) < 2 \lceil \log_2 n \rceil - 1$ for $n \geq 3$, respectively. The on-line version of a complete bipartite graph with $m \leq n$ was proved to be $m + 1 \leq \chi^*_r(K_{m,n}) \leq \min\{n+1,2m+1\}$ [8] and the on-line node ranking number of star graph is given by [13] as $\chi^*_r(S_n) = 3$ for $n \geq 3$.

There are several papers worked on on-line trees,
the best known on-line node ranking algorithm for
trees which has \( O(n^2) \) time complexity and
\( O(n) \) space complexity was given by [6]. [9]
proposed an online ranking algorithm for trees
using CREW PRAM model with \( O(n/\log^2 n) \)
processors. [10] provided an optimal online
ranking algorithm for stars and an \( O(n^3) \) time
algorithm for arbitrary trees.

This paper established offline node ranking
number for mesh and the corona of any two
trees which has the best known on-line node ranking algorithm for
trees using CREW PRAM model with \( O(n/\log^2 n) \)
processors. [10] provided an optimal online
ranking algorithm for stars and an \( O(n^3) \) time
algorithm for arbitrary trees.

This paper established offline node ranking
number for mesh and the corona of any two
trees which has the best known on-line node ranking algorithm for
trees using CREW PRAM model with \( O(n/\log^2 n) \)
processors. [10] provided an optimal online
ranking algorithm for stars and an \( O(n^3) \) time
algorithm for arbitrary trees.

## 2 Offline Results for Corona Graphs

Given graphs \( G \) and \( H \) with \( n \) and \( m \) vertices
respectively, the corona of \( G \) with respect to \( H \)
denoted as \( G \circ H \), is the graph with vertex set
\( V(G \circ H) = V(G) \cup \{n \text{ distinct copies of } V(H)\} \)
denoted as \( V(H_1) \cup V(H_2) \cup \cdots \cup V(H_n) \)
and the edge set \( E(G \circ H) = E(G) \cup \{n \text{ distinct copies of}\)
\( E(H)\} \cup E(H_1) \cup E(H_2) \cup \cdots \cup E(H_n) \)
\( \cup \{ (u, v) : u \in V(G), v \in V(H_i), \)
\( 1 \leq i \leq n \}. \) Figure 1 shows an example of graph
\( P_3 \circ P_2. \)

![Figure1: An example of graph \( P_3 \circ P_2. \)](image)

**Theorem 1:** Let \( G \circ H \) denote the corona of
\( G \) with respect to \( H \). Then the node ranking
number \( \chi(G \circ H) = \chi(G) + \chi(H) \).

**Proof:** First we show that \( \chi(G \circ H) \leq \chi(G) + \chi(H) \). Let \( f_1 \) and \( f_2 \) be optimal
node ranking labeling of \( G \) and \( H \) respectively.
Let \( V(G) = \{u_1, u_2, \ldots, u_n\} \) and
\( V(H) = \{v_1, v_2, \ldots, v_m\} \), \( V(G \circ H) = V(G) \cup \{v_i\} \)
which is the \( i \)-th copy of the \( j \)-th vertex of
\( H \), \( 1 \leq i \leq n, 1 \leq j \leq m \}. \) Define \( f \) as
\( f(u_i) = f_1(u_i) + \chi(H) \) for \( u_i \in V(G) \) and
\( f(v_{i,j}) = f_2(v_{i,j}) \) for \( v_{i,j} \in V(H) \), then \( f \) is a
node ranking labeling of \( G \circ H \) with maximum
rank \( \chi(G) + \chi(H) \), which implies
\( \chi(G \circ H) \leq \chi(G) + \chi(H) \).

Next we show that \( \chi(G \circ H) \geq \chi(G) + \chi(H) \).

Suppose to the contrary, that
\( \chi(G \circ H) < \chi(G) + \chi(H) \). Let \( g \) be an
optimal node ranking of \( G \circ H \).

**Case 1:** \( \forall 1 \leq i \leq n, 1 \leq j \leq m \), \( g(u_i) > g(v_{i,j}) \).

Then \( g(u_i) > \chi(H) \), \( \forall u_i \in V(G) \). Define a
labeling \( g' \) of \( G \) as \( g'(u_i) = g(u_i) - \chi(H) \),
then \( g' \) is a node ranking labeling of \( G \) with
maximum rank \( \chi(G \circ H) - \chi(H) < \chi(G) \),
which is a contradiction.

**Case 2:** \( g(u_i) < g(v_{i,j}) \), for some \( 1 \leq i \leq n, 1 \leq j \leq m \). Consider a labeling \( h \) of \( G \circ H \),
which exchange the label of \( u_i \) with the
minimum label of \( v_{i,j} \) for those \( v_{i,j} \) having
label greater than \( u_i \). That is, if \( y \) is the
minimum label of \( V(H) \) such that \( g(u_i) < y \),
then for all vertices \( v_{i,j} \) with \( g(v_{i,j}) = y \), let
\( h(v_{i,j}) = g(u_i) \), \( h(u_i) = y \), and \( h(x) = g(x) \)
or otherwise. Repeat the same process on \( h \) until
\( h(u_i) > h(v_{i,j}) \) for all \( 1 \leq i \leq n, 1 \leq j \leq m \). Note
that for each exchange, since it starts from the
smallest label, the order of the ranks are not
changed in graph \( H_1 \), and for \( h(u_i) = h(u_j) \)
we must have some \( j \) that \( g(v_{i,j}) = g(u_j) \), since \( g \)
is a node ranking labeling of graph \( G \circ H \), there
must be a \( w \) on \( u_i - u_j \) path with
\( h(w) > h(u_i) = h(u_j) \), which implies \( h \) is a
node ranking labeling of \( G \circ H \). Since the
maximum label of \( h \) is the same as the maximum
corona of \( g \), which lead us back to case 1 and
produce a contradiction.

Let \( S_{n,m} \) be the graph which contains a \( n \)-cycle
and each cycle vertex has \( m \) hairs. Since the
graph has the shape of sun, we named it as “sun graph”. Figure 2 shows an example of graph
\( S_{5,2}. \)

![Figure2: An example of graph \( S_{5,2}. \)](image)

Since the sun graph \( S_{n,m} \) may be viewed as
\( C_n \circ K_m \), following corollary comes directly from
Theorem 1 by knowing that \( \chi(C_n) = \lceil \log_2 n \rceil + 1 \)
(from [1]) and \( \chi(K_m) = 1 \).
Corollary 1: Let $S_{n,m}$ be the sun graph of order $n(m+1)$. Then the node ranking number $\chi_2(S_{n,m}) = \lceil \log_2 n \rceil + 2$.

3 Offline Results for Mesh Graphs

In this section, we proposed an off-line ranking algorithm for mesh graphs $P_n \times P_n$. The algorithm produced a node ranking labeling of $P_n \times P_n$ for odd $n$ and $n \geq 15$, for even $n$ and $n \geq 18$. Figure 3 shows an example of the graph $P_n \times P_n$.

![Figure 3: An example of graph $P_n \times P_n$.](image)

Algorithm **Offline_Ranking_Mesh**

**Input:** $n, n \geq 15$ if $n$ is odd and $n \geq 18$ if $n$ is even. Each vertex is denoted as $v_{i,j}$ for $1 \leq i, j \leq n$.

**Output:** An off-line rank assignment of $P_n \times P_n$.

**Method:**

1. **Step 1:**
   - If $n$ is odd, then
     - $x = y = 0$
     - $x = n - 1$, $y = 0$
     - $x = 0$, $y = n - 1$
     - $x = n - 1$, $y = n - 1$
   - Else if $n$ is even, then
     - $x = y = 0$
     - $x = n - 1$, $y = 0$
     - $x = 0$, $y = n - 1$
     - $x = n - 1$, $y = n - 1$

2. **Step 2:**
   - For $x = 1$ to $\lfloor n/2 \rfloor$
     - Dividing the graph by $x + y = n$.
   - For $x = \lceil n/2 \rceil$ to $n$
     - Dividing the graph by $x - y = 1$.

3. **Step 3:**
   - For $x = 2$ to $\lfloor n/2 \rfloor$
     - Dividing the graph by $y = x + 1$.
   - For $x = \lceil n/2 \rceil + 1$ to $n - 1$
     - Dividing the graph by $y = x + 1$.

4. **Step 4:**
   - In the $\alpha_1$ graph, $i = \lfloor n/2 \rfloor - 1$; call Alpha $(i)$; use the label for corresponding vertices in $\alpha_2$ to $\alpha_4$.
   - $\alpha_1$ means the number of nodes on the edge of the subgraph $\alpha_1$.
   - Else if $n$ is even and $n = 0 \mod 4$, then
     - Step 1: If $x + y = 0$ then $y[x] = 1$.
     - For $x = \lfloor n/2 \rfloor + 3$ to $n/2 + 4$
       - Dividing the graph by $x = x + y = n/2 + 5$.
     - For $x = n/2 + 3$ to $n/2 + 4$
       - Dividing the graph by $x = x - y = 1$.
     - For $x = n/2$ to $n/2 + 1$
       - Dividing the graph by $y = x = n + 1$.
     - For $x = n/2$ to $3n/4 - 2$
       - Dividing the graph by $y = x = 1$.
     - For $x = n/2 - 3$ to $3n/4 - 2$
       - Dividing the graph by $x + y = 3n/2 - 3 = u_0$.

5. **Step 5:** In the $\alpha_5$ graph, $i = n/4$; call Alpha $(i)$; In the $\alpha_6$ graph, $j = \lceil (3n - 12)/8 \rceil$; call Alpha $(j)$;
   - In $\delta_1$ and $\delta_2$, there are lines of vertices denoted as $e_1, e_2, \cdots, e_i$, such that
\[ x + y = u_i + 2c, \quad 1 \leq c \leq \left(\frac{u_0 - u_i}{2}\right) \]

where \( l = 3n/4 - 4\frac{(3n - 12)}{8} \). Let \( j = \lceil l/2 \rceil \).

Dividing \( \beta_1 \) and \( \beta_2 \) by \( e_j \) and \( e_{j+1} \), respectively.

Let \( u_x = x + y \) for the vertices \( v_{r,s} \) on line \( e_j \) and \( u_{y} = x + y \) for the vertices \( v_{r,s} \) on line \( e_{j+1} \).

The graph \( \beta_1 \) is divided into two subgraphs \( \beta_{11}, \beta_{12} \) and the graph \( \beta_2 \) is divided into two subgraphs \( \beta_{21}, \beta_{22} \) as shown in Figure 8.

![Figure 8: An example of graphs \( \beta_1, \beta_1, \beta_2, \beta_2 \).](image)

**Step 5:**
- In \( \beta_{11} \), Dividing the graph by \( y - x = n/2 + 2(\lceil n/2 \rceil - 1) = v_1 \).
- In \( \beta_{12} \), Dividing the graph by \( y - x = n/2 + 2((n - 4)/16) + 1 = v_2 \).

**Step 6:**
- Let \( p = (u_0 - u_i)/2 - 1 \); \( q = (v_0 - 1)/2 \); call \( \alpha(0, q) \);
- Let \( p = (u_0 - u_i)/2 - 1 \); \( q = (v_0 - 3)/2 \); call \( \beta(p, q) \).

**Step 7:**
- In \( \alpha_1 \), \( i = (u_0 - v_i)/2 - 2 \); call \( \alpha(i) \);
- In \( \beta_{11} \), let \( p = (u_0 - u_i)/2 - 1 \); \( q = (v_1 - u_0)/2 - 1 \); call \( \beta(p, q) \);
- In \( \alpha_2 \), \( i = (u_0 - u_i)/2 - 2 \); call \( \alpha(i) \);
- In \( \beta_{12} \), let \( p = (u_0 - u_i)/2 - 1 \); \( q = (v_1 - v_0)/2 - 1 \); call \( \beta(p, q) \);

**Else** // the graph is divided into two isomorphic subgraphs \( \alpha_0, \alpha_2 \).

**Step 1:** If \( x + y = y \) Then \( \text{array}[y][x] = 1 \).

**Step 2:** For \( x = \lceil n/4 \rceil + 3 \) to \( n/2 + 3 \) do
- Dividing the graph by \( x + y = n/2 + 4 \).
- For \( x = \lceil n/4 \rceil + 3 \) to \( n/2 + 1 \) do
  - Dividing the graph by \( x - y = 1 \).

**For** \( x = n/2 \) to \( n/2 + 1 \) do
- Dividing the graph by \( x + y = n + 1 \).

**For** \( x = n/2 \) to \( 3 \lceil n/4 \rceil \) do
- Dividing the graph by \( y - x = 1 \).

**For** \( x = n/2 - 2 \) to \( 3 \lceil n/4 \rceil \) do
- Dividing the graph by \( x + y = 3n/2 - 2 = u_i \).

// The graph is divided into two isomorphism subgraphs \( \alpha_0, \alpha_2 \) which is the same as in Figure 5.

**Step 3:** In the \( \alpha_0 \) graph, Dividing the graph by \( x + y = n/2 + 2((n - 2)/8) = u_0 \). (Repeat the same processes for the corresponding vertices in \( \alpha_2 \)).

// The graph is divided into two subgraphs \( \epsilon_1, \epsilon_2 \) as shown in Figure 6.

**Step 4:** Dividing \( \epsilon_1 \) and \( \epsilon_2 \) by \( x - y = 1 \) and \( y - x = n/2 = v_i \), respectively.

// The graph \( \epsilon_1 \) is divided into two subgraphs \( \alpha_1, \alpha_2 \) and the graph \( \epsilon_2 \) is divided into two subgraphs \( \beta_1, \beta_2 \) as shown in Figure 7.

**Step 5:** In \( \alpha_{11} \), \( i = n/4 \); call \( \alpha(i) \);
- In \( \alpha_{12} \), \( i = (\lceil 3n/4 - 1/2 \rceil) \); call \( \alpha(i) \);
- In \( \beta_{11} \), there are lines of vertices \( e_1, e_2, \ldots, e_j \) such that \( x + y = u_i + 2c \), where \( 1 \leq c \leq (u_0 - u_i)/2 - 1 \) and \( j = \lceil (n + 2)/8 \rceil + (n/2)/4 - 3 \).

**Step 6:** Dividing the graph \( \delta_{11} \) by \( y - x = n/2 + 2((n + 2)/8) = v_i \).
- In \( \delta_{12} \), Dividing the graph by \( y - x = n/2 + 2((n + 2)/16) = v_0 \).

**Step 7:**
- In \( \alpha_{11} \), let \( i = (u_0 - v_i)/2 - 2 \); call \( \alpha(i) \);
- In \( \alpha_{12} \), let \( p = (u_0 - u_i)/2 - 1 \); \( q = (v_0 - 1)/2 \); call \( \beta(p, q) \);
- In \( \beta_{11} \), let \( p = (u_0 - u_i)/2 - 1 \); \( q = (v_1 - v_0)/2 - 1 \); call \( \beta(p, q) \);
- In \( \beta_{12} \), let \( p = (u_0 - u_i)/2 - 1 \); \( q = (v_1 - v_0)/2 - 1 \); call \( \beta(p, q) \);

**End of Algorithm Offline_Ranking_Mesh_Graph**

**Procedure Alpha** (int \( i \))
- If \( i \geq 3 \) then \{ \[ p = \lceil i/3 \rceil \]; \( q = i - p - 1 \); \}
\[ p \text{ means the number of nodes of the dividing line } P. \]
\[ p_1 = p + p; \]
Dividing the graph by \( I : y - x = 2p_1 + 2c - 1 \).
Dividing the graph by \( P : x + y = 2p_1 + 2c + 1 \).

\[ \begin{align*}
\text{Dividing the graph by } P_1: & \quad x + y + 1 = p_1 + 1 \\
\text{Dividing the graph by } P_2: & \quad y - x - 1 = p_1 - 1 \\
\text{Dividing the graph by } P_3: & \quad x + y + 1 = 2p_1 + 2c - 1
\end{align*} \]

The graph \( \alpha_0 \) is divided into three subgraphs \( \alpha_{11}, \alpha_{12}, \alpha_{13} \) as shown in Figure 10.

**Procedure Beta ( int \( p \), int \( q \))**

If \( (q \geq 3) \) Then 

In \( \beta_0 \), there are dividing lines : \( a_1, a_2, \ldots, a_q \) as shown in Figure 12.

**Procedure Gamma ( int \( p \))**

If \( (p \geq 3) \) Then 

In \( \gamma_0 \), there are dividing lines : \( b_1, b_2, \ldots, b_p \) as shown in Figure 13.

End of Procedure Beta

End of Procedure Gamma

End of Procedure Alpha
Dividing the graph by \( b_{p/2+1} \).

\( p = p/2 ; \)

}\)

Else If \( (p = 2) \) Then Rank the nodes using 2 to 5.
Else Rank the nodes individually with 2 and 3.
call Gamma \( (p) \);

\textbf{End of Procedure Gamma}

\textbf{Theorem 2:} The labeling given by Algorithm Offline_Ranking_Mesh is a node ranking labeling of \( P_n \times P_n \) for odd \( n \) and \( n \geq 15 \), for even \( n \) and \( n \geq 18 \).

\textbf{Proof:} In the algorithm, we always assign greater rank for the dividing vertices, and on every path between two nodes of the same rank there is at least one dividing line, so the labels given by the algorithm must follow the node ranking condition for \( P_n \times P_n \).

\textbf{Theorem 3:} The maximum rank given by Algorithm Offline_Ranking_Mesh for odd \( n \) and \( n \geq 15 \) is:

\[
\begin{align*}
& n + \left\lfloor \frac{n}{2} \right\rfloor + \left\lfloor \frac{n+1}{6} \right\rfloor + \left\lfloor \frac{n+1}{6} \right\rfloor + 8\left(\frac{n-1}{12}\right) \\
& -10 + x; \quad \text{and for even } n, \quad n \equiv 0 \mod 4 \quad \text{and} \quad n \geq 20 \\
& \text{is } n + n/4 + \left\lfloor \frac{n-2}{8} \right\rfloor + 1 + \max \left\{ 3\left\lfloor \frac{n-4}{2} \right\rfloor \right. \\
& \left. + \left\lfloor \frac{n}{8} \right\rfloor - 5 \right\} \quad \text{and for even } n, \quad n \not\equiv 0 \mod 4 \quad \text{and} \\
& n \geq 18 \text{ is } n + \left\lfloor \frac{n}{4} \right\rfloor + \left\lfloor \frac{n-2}{8} \right\rfloor - 1 \\
& + \max \left\{ 3\left\lfloor \frac{n}{4} \right\rfloor - 2 + \left\lfloor \frac{n}{8} \right\rfloor \right. \\
& \left. + 2\left\lfloor \frac{n}{4} \right\rfloor + \left\lfloor \frac{n+3}{12} \right\rfloor \right. \\
& \left. + \left\lfloor \frac{n+3}{12} \right\rfloor + 8\left(\frac{n-3}{24}\right) \right\} - 9 + x \}, \quad \text{where} \\
& \begin{cases} 
0 & \text{if } y = 0, 2 \\
2 & \text{if } y = 4 \\
4 & \text{if } y = 6, 8 \\
5 & \text{if } y = 10 
\end{cases} \quad \text{for } y = n - \left(3 + 12\left(\frac{n-3}{12}\right)\right).
\]

Theorem 3 gave us an upper bound for the node ranking number of a mesh \( P_n \times P_n \) for \( n \geq 17 \). For \( n \leq 16 \), Table 1 shows the node ranking number of \( P_n \times P_n \) which is obtained by a brute force computer testing, and is denoted by the formula that represents the relation between \( n \) and \( \chi_r(P_n \times P_n) \).

\begin{table}[h]
\centering
\begin{tabular}{|c|c|}
\hline
\( n \) & \( \chi_r(P_n \times P_n) \) \\
\hline
\( 1 \leq n \leq 7 \) & \( 2n - 1 \) \\
\( 8 \leq n \leq 10 \) & \( 2n \) \\
11 & \( 2n + 1 \) \\
12 & \( 2n \) \\
13 & \( 2n + 2 \) \\
14 & \( 2n + 1 \) \\
15 & \( 2n + 3 \) \\
16 & \( 2n + 2 \) \\
\hline
\end{tabular}
\caption{\( \chi_r(P_n \times P_n) \) for \( n \leq 16 \).}
\end{table}

\section{Online Results}

In this section, we proposed an on-line ranking algorithm for sun graphs \( S_{n,m} = C_n \odot K_m \), and analyze both time and space complexities of our algorithm. Notice that there are \( n(m+1) \) vertices in \( S_{n,m} \). Let \( v_i \) to \( v_{n(m+1)} \) represent the order of the coming vertices.

\textbf{Algorithm Online_Ranking_Sun_Graph}

We use structure array Sun[ ] to record the information of each node, and use value_record[] as the counter of the index value’s appearance.

\textbf{Input:}

\( n \): the number of vertices on the cycle \( S_{n,m} \).
\( m \): the number of vertices on the hairs of \( S_{n,m} \).
\( v_i \) for \( 1 \leq i \leq m(m+1) \) the vertices together with the adjacent vertices \( v_j \) for \( 1 \leq j < i \).

\textbf{Output:}

An online node ranking labeling of \( S_{n,m} \).

\textbf{Method:}

\begin{enumerate}
\item Set the occurrence record to zero except rank 1 (which is set to 2)
\item Applying online tree algorithm given in [5] to loop through each adjacent path, go as long as possible, to find the max value that has occurred at least twice
\item Find minimum available rank \( k \) that is greater than \( j \)
\item Set \( rank(v_i) = k \)
\end{enumerate}

\textbf{End of Algorithm Online_Ranking_Sun_Graph}

\textbf{Theorem 4:} Algorithm Online_Ranking_Sun has \( O\left(\frac{mn}{n}\right) \) time complexity and \( O(mn) \) space complexity.

\textbf{Proof:}

First we analyze the time complexity of the algorithm. Since a Sun graph with \( (m+1)n \) vertices has exactly \( (m+1)n \) edges, each time when insert a new node \( v_i \), the looping for finding max \( j \) used online tree algorithm which takes \( O(i^2) \) times, so after \( (m+1)n \) vertices inserted, the whole process will be at most \( 1^2 + 2^2 + \ldots + ((m+1)n)^2 = O\left(\frac{mn}{n}\right)^2 \) times. For the space complexity, we use structure array with \( (m+1)n \) elements to store the node plus one array to record the rank occurrence, the total space complexity then is \( O(mn) + O(mn) = O(mn) \). □
Theorem 5: For a sun graph $S_{n,m}$ with $n \geq 7$ is $\lceil \log_2 n \rceil + 2 \leq \chi_r(S_{n,m}) \leq \lceil \log_2 n \rceil + 6m$.

Proof:
Since the online node ranking number is at least as much as the offline node ranking number, we know that $\lceil \log_2 n \rceil + 2 \leq \chi_r(S_{n,m})$. Let $S_{n,m}$ be the graph which contains a $n$-cycle $v_i$ for $1 \leq i \leq n$ and each cycle vertex has $m$ hairs denoted as $u_{i,j}$ for $1 \leq i \leq n, 1 \leq j \leq m$. If we use adversary analysis, for the best we can do, saving rank 1 for the vertices which are known as hair, assume that the vertices of $S_{n,m}$ are coming as follows:

- $v_1, v_2, v_1, v_4$ coming in first, so they get labels $2,3,2,4$.
- For $3 \leq i \leq n-3$, all $u_{i,j}$ for $1 \leq j \leq m$ comes before $v_{i+2}$.
- Then comes $u_{n-2,j}$ for $1 \leq j \leq m$ and $u_{i,j}$ for $i=1,2, n-1$ and $1 \leq j \leq m$.
- At last, $v_{n}$ then $u_{n,j}$ for $1 \leq j \leq m$.

In this coming order, before $v_{n}$ comes in, no vertex can be recognized as end vertex, hence online tree algorithm worked for this case which is the worst case and our algorithm guarantee the maximum rank is no more than $\lceil \log_2 n \rceil + 6m$, hence we have $\chi_r(S_{n,m}) \leq \lceil \log_2 n \rceil + 6m$, which complete the proof.

5 Conclusion

There are several papers discussed the node ranking problem. In this paper, we presented off-line node ranking number for the corona graphs and gave labeling algorithm for mesh. We also gave on-line algorithm for sun graphs. Both time and space complexities of our algorithms are analyzed.

References