We obtain a new approach to calculate the number of spanning trees for any simple undirected graphs on the basis of the singular value of the incidence matrix. By this result we obtain J program for computing the complexity of graphs.
Complexity of Graph and its J Program

Yoshiko Takenaka*
Juichiro Takeuchi**
Katsuo Matsuoka***

Summary—We obtain a new approach to calculate the number of spanning trees for any simple undirected graphs on the basis of the singular value of the incidence matrix. By this result we obtain J program for computing the complexity of graphs.

Key words: J program, singular value, complexity of graph

1 INTRODUCTION

$G=(V,E)$ is a simple undirected graph with a vertex set $V=\{v_1,v_2,...,v_n\}$ and an edge set $E=\{e_1,e_2,...,e_m\}$. $K(G)$ is the complexity of graph $G$ i.e. the number of spanning tree of $G$. By giving an arbitrary orientation to each edge of $G$, we define the

$$
E, = \begin{cases} 
1 & \text{if } v_i \text{ is the initial vertex of } e_j \\
-1 & \text{if } v_i \text{ is the terminal vertex of } e_j \\
0 & \text{otherwise}
\end{cases}
$$

$E_i$ is the matrix obtained from $E$ by removing the $i$-th row. It is clear that

$$
\text{rank } E = \text{rank } E_i = n - 1
$$

for $i \in \{1,2,...,n\}$.

*竹中庸子, 建部義大大学数学教室 (〒223-8521 横浜市港北区日吉4-1-1) : Dept. of Economics, Keio Univ., Hiyoshi, Kohoku-ku, Yokohama 223-8521, Japan. **竹内秀一郎, 建部義大大学理工学部 : Dept. of Engineering Science, Keio Univ., Hiyoshi, Kohoku-ku, Yokohama 223-8521, Japan. e-mail:takeuchi@ae.keio.ac.jp. ***訳内雄, 日本大学経済学部, Dept. of Economics, Nihon Univ., Tokyo, Japan. [Received Feb. 29, 2000]
Also we denote $A_i$ the matrix obtained from the adjacent matrix $A$ of $G$ by removing the $i$-th row and $i$-th column. It means $A_i$ is the adjacency matrix of $G - v_i$. By the same definition as $A_i$, $D_i$ is the degree matrix $D = (\delta_y d(y_i))$ of $G$, where $d(v_i)$ is the degree of vertex $v_i$. We put

$$L = D - A$$

and call $L$ the combinatral Laplacian of $G$. Matrix-tree Theorem states that

$$\kappa(G) = \det L_{ij}$$

where $L_{ij}$ is the $(i,j)$ cofactor of $L$ and $i, j \in \{1, 2, ..., n\}$.

Moreover between the $(0, 1, -1)$-incidence matrix and the combinatral Laplacian, there is the relation

$$E' E = L.$$

$A, D$ is the adjacency matrix and the degree matrix of $G$ respectively, then

$$E_i' E_i = L_{ii} = D_i - A_i,$$

where $L_{ii}$ is the $(i,i)$ cofactor of $L$ and $i \in \{1, 2, ..., n\}$.

Consequently, $E'E$ and $E_i'E_i$ are independent of the orientation given to $G$ and we have

$$\kappa(G) = \det E_i'E_i = \det(D_i - A_i)$$

2 SINGULAR VALUE

Now, we introduce the singular value $\text{Sing} (G)$ of graph $G$. Let $E$ be a $(0, 1, -1)$-incidence matrix of $G$, then $E'_i E_i$ is a regular and real symmetrical matrix. $E_i$ is positive definite and $E'_i E_i$ is diagonalizable. We give the following definition in the same manner as the definition of the spectrum of a graph.
DEFINITION

$E$ is the $(0,1,-1)$-incidence matrix of $G$ and $\lambda_1, \lambda_2, \ldots, \lambda_s$ with $\lambda_1 > \lambda_2 > \ldots > \lambda_s > 0$ are the eigenvalues of $E_iE_i$ and $m_1, m_2, \ldots, m_s$ are their multiplicities.

We call $u_i = \sqrt{\lambda_1}, u_2 = \sqrt{\lambda_2}, \ldots, u_s = \sqrt{\lambda_s}$ as the singular values of $E_i$ and denote the singular values of $G$ with respect to $E_i$ by

$$\text{Sing}_i(G) = \left( \frac{\mu_1}{m_1}, \frac{\mu_2}{m_2}, \ldots, \frac{\mu_s}{m_s} \right)$$

In particular, if $\text{Sing}_i(G)$ is unique independently of $i$, we call it as the singular value of $G$ and denote it by $\text{Sing}(G)$.

For a regular graph $G$ and any vertex $v$ of $G$, $G \cdot v$ is isomorphic, therefore there exists $\text{Sing}(G)$. For the $(0,1,-1)$-incidence matrix $E$ of $G$, $E_iE_i$ is diagonalizable and

$$\det E_iE_i = \prod_{i=1}^{s} (u_i^2)^{m_i}$$

Then we obtain the following new method to calculate the complexity of a graph.

**THEOREM** Let $G$ be a graph and let

$$\text{Sing}(G) = \left( \frac{\mu_1}{m_1}, \frac{\mu_2}{m_2}, \ldots, \frac{\mu_s}{m_s} \right)$$

then,

$$\kappa(G) = \prod_{i=1}^{s} (u_i^2)^{m_i}$$

3 EXAMPLES

By this theorem, we calculate the singular values of several special families of graphs and write down their complexities, which equal to the well known results.

1. For the complete graph with $n$ vertices $K_n$,

$$\text{Sing}(K_n) = \left( \frac{\sqrt{n}}{n-2}, 1 \right)$$
Complexity of Graph and its J Program

(TAKENAKA, TAKEUCHI, MATSUOKA)

So we have

$$\kappa(K_n) = n^{n-1}$$

2. For the circuit graph of length \( n \) \( C_n \),

$$\text{Sing}(C_n) = \left( \sqrt{2 - 2 \cos \frac{(n-1)\pi}{n}} \quad \sqrt{2 - 2 \cos \frac{(n-2)\pi}{n}} \quad \cdots \quad \sqrt{2 - 2 \cos \frac{\pi}{n}} \right) .$$

So we have

$$\kappa(C_n) = \prod_{k=1}^{n-1} (2 - 2 \cos \frac{k\pi}{n})$$

3. For the wheel graph with \( n \) vertices \( W_n \),

$$\text{Sing}(W_n) = \left\{ \begin{array}{ll}
\left( \sqrt{3 - 2 \cos \frac{(l-1)\pi}{l}} \quad \sqrt{3 - 2 \cos \frac{(l-2)\pi}{l}} \quad \cdots \quad \sqrt{3 - 2 \cos \frac{\pi}{l}} \quad 1 \right) & \text{if } n = 2l, \\
\left( 5 \quad \sqrt{3 - 2 \cos \frac{(l-1)\pi}{l}} \quad \sqrt{3 - 2 \cos \frac{(l-2)\pi}{l}} \quad \cdots \quad \sqrt{3 - 2 \cos \frac{\pi}{l}} \quad 1 \right) & \text{if } n = 2l + 1.
\end{array} \right.$$  

So we have

$$\kappa(W_n) = \left\{ \begin{array}{ll}
\prod_{k=1}^{l-1} \left( 3 - 2 \cos \frac{2k\pi}{2l-1} \right) & \text{if } n = 2l, \\
5 \prod_{k=1}^{l-1} \left( 3 - 2 \cos \frac{k\pi}{l} \right) & \text{if } n = 2l + 1.
\end{array} \right.$$  

4. For the complete bipartite graph \( K_{r,s} (r \geq s) \)

$$\text{Sing}(K_{r,s}) = \left\{ \begin{array}{ll}
\left( \begin{array}{c}
1 \\
s
\end{array} \right) & \text{if } r = 1, \\
\left( \begin{array}{c}
\sqrt{r + s + \sqrt{(r + s)^2 - 4s}} \\
2 \\
\sqrt{s} \\
\sqrt{s} \\
\sqrt{r} \\
1
\end{array} \right) & \text{if } r \geq 2.
\end{array} \right.$$
So we have

$$\kappa(K_{r,s}) = r^{r-1}s^{s-1}$$

4 COMPLEXITY of $G(20,30)$

By this theorem we obtain $J$ program for computing the complexity of graphs. (cf. Appendix)

[Examples]

$$G = (V, E), |V| = 20, |E| = 30$$

We label the number $1, 2, \ldots, 20$ to all vertices of graph $G$ and indicate edges by two vertices number. We input the number series corresponding to all edges of $G$, no particular order observed. The important point of this program is that input is a simple number series and not a matrix. Then we obtain $\operatorname{Sing}(G)$ and $\chi(G)$ of graph $G(20,30)$ as output.

$$X = 12, 23, 34, 45, 15, 16, 28, 310, 412, 514,$$

$$67, 78, 89, 910, 1111, 1213, 1314, 1415, 615, 716,$$

$$917, 1118, 1319, 1520, 1620, 1617, 1718, 1819, 1920$$

$$\chi(G) = 5,184,000$$
### Complexity of Graph and its J Program

| 0.15 | 7.6 | -0.2 | R | 7.7 | 1.0 | 0.357 | 2.0 | 0.001 | 2.6 | 0.125 | 3.5 | 2.700 |
| 0.15 | 7.6 | -0.2 | R | 7.7 | 1.0 | 0.357 | 2.0 | 0.001 | 2.6 | 0.125 | 3.5 | 2.700 |
| 0.15 | 7.6 | -0.2 | R | 7.7 | 1.0 | 0.357 | 2.0 | 0.001 | 2.6 | 0.125 | 3.5 | 2.700 |
| 0.15 | 7.6 | -0.2 | R | 7.7 | 1.0 | 0.357 | 2.0 | 0.001 | 2.6 | 0.125 | 3.5 | 2.700 |
| 0.15 | 7.6 | -0.2 | R | 7.7 | 1.0 | 0.357 | 2.0 | 0.001 | 2.6 | 0.125 | 3.5 | 2.700 |
| 0.15 | 7.6 | -0.2 | R | 7.7 | 1.0 | 0.357 | 2.0 | 0.001 | 2.6 | 0.125 | 3.5 | 2.700 |
| 0.15 | 7.6 | -0.2 | R | 7.7 | 1.0 | 0.357 | 2.0 | 0.001 | 2.6 | 0.125 | 3.5 | 2.700 |
| 0.15 | 7.6 | -0.2 | R | 7.7 | 1.0 | 0.357 | 2.0 | 0.001 | 2.6 | 0.125 | 3.5 | 2.700 |
| 0.15 | 7.6 | -0.2 | R | 7.7 | 1.0 | 0.357 | 2.0 | 0.001 | 2.6 | 0.125 | 3.5 | 2.700 |
| 0.15 | 7.6 | -0.2 | R | 7.7 | 1.0 | 0.357 | 2.0 | 0.001 | 2.6 | 0.125 | 3.5 | 2.700 |
| 0.15 | 7.6 | -0.2 | R | 7.7 | 1.0 | 0.357 | 2.0 | 0.001 | 2.6 | 0.125 | 3.5 | 2.700 |

### 19 eigen values and their eigenvectors

| 1.243072 | 0.794837 | 0.806521 | 1.306 | 0.075393 | 0.279394 | 0.145958 | 0.257772 | 0.143378 |
| 0.60551218 | 0.206427 | 0.216179 | 0.157346 | 0.141513 | 0.104826 | 0.202972 | 0.192534 |
| 0.410367 | 0.704957 | 0.218971 | 0.174843 | 0.213753 | 0.255849 | 0.071734 | 0.072030 |
| 0.107359 | 0.159932 | 0.032650 | 0.124954 | 0.121694 | 0.119635 | 0.027072 | 0.027072 |
| 0.0277355 | 0.9264 | 0.202207 | 0.299431 | 0.243153 | 0.214943 | 0.204793 | 0.204793 |
| 0.174238 | 0.270957 | 0.328981 | 0.1213577 | 0.015346 | 0.143148 | 0.046247 | 0.046247 |
| 0.270059 | 0.010436 | 0.206457 | 0.385425 | 0.241646 | 0.342266 | 0.026235 | 0.026235 |
| 0.173219 | 0.270077 | 0.229941 | 0.089456 | 0.133913 | 0.193939 | 0.282928 | 0.282928 |
| 0.157657 | 0.204078 | 0.315283 | 0.273473 | 0.151449 | 0.284697 | 0.065214 | 0.065214 |
| 0.160589 | 0.127314 | 0.052731 | 0.164495 | 0.004841 | 0.232879 | 0.026432 | 0.026432 |
| 0.174207 | 0.270059 | 0.206457 | 0.385425 | 0.241646 | 0.342266 | 0.026235 | 0.026235 |
| 0.10459128 | 0.204078 | 0.315283 | 0.273473 | 0.151449 | 0.284697 | 0.065214 | 0.065214 |

### 17 eigen values and their eigenvectors
Appendix: J Program for Computing Complexity

J is a general-purpose programming language available as shareware on a wide variety of computers (http://www.jsoftware.com). The characteristics of J is that we can handle vectors, matrices, and other arrays as single entities like a pocket calculator. J has many kinds of convenient functions for instance, matrix operations including transpose or inverse matrix, multiple regression, outer products, operations with complex numbers, Taylor expansions, approximations by polynomials, decomposition into prime numbers and GCD and LCM. We can easily get J freeware from the following address ftp://take.soc.ae.keio.ac.jp/take/jlexperlfdl/jfw.exe.

Subroutines: (NB. Denotes Comments)

NB. Write to Screen
print=. (1!:2)&2

NB. Matrix Product or Inner Product
mp=.*

NB. Select eigen vectors corresponding to nonzeo eigen values
nse1=. 1e-5&<@(0&{)@{.#h1).

NB. Convert a vector of locations to a matrix form
mat=. 3*:1(=<1 (x..(m#2)#i.n))((>./y.),n=.y.,y.n)#$0'

NB. Replace the first "1" in each column by "_1"
minus=. 3*:1(=<1 t.,i.#t.?(1:1)"1(y)).

NB. Jacobi's method for eigenvalues and vectors
NB. form [tolerance] jacobi mat
NB. default tolerance 1e_5
NB. returns: eigenvalues;eigenvectors
 jacobi=3:0
1e_5 jacobi y.

r=. y.
ir=. i.#r
q=. id=. =ir
diag=. (<0 1)&l:
len=. +/&(.*""
imax=. i.>.
utm=. ,.<~ir
mp=. */.
ndx=. (;~ir
perm=. 0 0{;}l.;1 1{;
sign=. _1: ^<&0
count=. 0
while.
   big=. imax utm * l,.r
   ind=. big { ndx
   p=. -ind { r
   (x. < |p) *. big > 0
do.
    u=: .-./ (>ind) { diag r
    v=: len p,u
    cos=: %: (v+|u) % +:v
    sin=: (sign u) * p % +: v * cos
    s=: ((cos,-sin),sin,cos) (perm >ind) } id
    r=: s mp r mp |:a
    q=: q mp |:s
    count=: >:countW
end.
    r=: diag r
    inx=: ¥:r
    (inx(r);inx("1 q
)

Main Program for Calculating Complexity of a Graph
Complex=:3:0
a=:}. minus print Mtrx=:mat y.
b=:a mp (|:a)
c=:jacobi b
n=:$.print Vec=:nsel c
print*/print Eig=:n {. c

An Example:
NB. data
    x=:1 2 3 4 1 4 1 3
NB. Execution form
    Complex x
NB. Graph matrix
    1 0 0 1 1
    1 1 0 0 0
    0 1 1 0 1
    0 0 1 1 0
NB. Eigen vectors
    _0.4082494 _0.7071062 0.5773503
    _0.8164966 _1.248312e6 0.5773503
    _0.4082472 0.7071074 0.5773503
NB. Eigen values
    4 2 1
NB. Complexity
    8
(mailing address : takeuchi@ae.keio.ac.jp)
REFERENCES