アルゴリズム 81－9
（2001．11．27）

An Optimal File Transfer on a Path Network with 2-level Arc Cost and Positive Demands

Yoshihiro Kaneko
Faculty of Engineering, GIFU University, 1-1Yanagito, GIFU, 501-1193, JAPAN
E-mail:kaneko@info.gifu-u.ac.jp

Summary: A problem of obtaining an optimal file transfer on a file transmission net N is to consider how to distribute, with the minimum total cost, copies of a certain file of information from source vertex to others on N by the respective vertices' copy demand numbers. The problem is NP-hard for a general file transmission net. Some class of N on which polynomial time algorithm to solve the problem has been known. So far, we assumed that N has a linear function as an arc cost. In this report, we suppose that N has a directed path graph structure with its arc costs being 2-level step functions. As a result, we show that the problem can be solved in $O(n^2)$, where n is the number of vertices.

Key Words: optimal file transfer, two-level step arc cost, directed path network, $O(n^2)$ algorithm

1. Introduction

The problem of locating facilities in a network system with existing demand while optimizing a certain objective function is well-known as a network location problem. As computer networks become larger but more common such as the Internet, the study of network location problem becomes more important. In fact, a number of relevant problems have been studied over the last several years[1－6].

In this report, we define and consider a file transmission net N. Our file transfer problem is to determine how the numbers of copies are to be transmitted as well as duplicated so as to meet the demands within N. The cost to evaluate for such file transfer is the sum of transmission and duplication. We say a file transfer with the minimum cost is optimal. Our file transfer problem is to obtain an optimal file transfer on a given net N.

This file transfer problem is quite different from the data transfer problem[3] and the scheduling file transfer problem[4]. Both of them deal with optimization based on time. Some similar problems are known such as the file allocation problem[5] and the file assignment problem[6] which differ from our problem that restrict vertex copying. Previously, we assumed that arc costs functions were linear for N[7-10]. However, from a practical viewpoint, we should also consider the case that each arc cost is a step function. This version is more difficult due to the nonlinearilty of the arc cost functions. As shown before, in general, this version is NP-hard [11]. So far, we restricted that N has a directed path graph structure with a two-level arc cost and each vertex demand being one. As a result, we proposed an $O(n^2)$ algorithm to obtain an optimal file transfer, where n is the number of vertices in N [13]. In this report, we propose an $O(n^2)$ algorithm to obtain an optimal file transfer on N with each vertex demand being positive integer.

In the rest of the report, we proceed as follows. In Section2, we give some necessary definitions for our file transfer problem. In Section3, after defining some operations to obtain an optimal file transfer such as copy exchange, forward operation, backward operation, and so forth, we propose an algorithm to obtain an optimal file transfer with its analysis.

2. Preliminaries

In this section, we define some terms in order to formulate a problem of obtaining an optimal file transfer on a file transmission net. For basic terminologies such as vertex, arc, path and so forth, on graph theory, refer to those in [12]. Let Z^+ denote the set of all positive integers and

let $Z_0^+ = Z^+ \cup \{0\}$. In the following, an arc, a path and a graph indicate a directed arc, a directed path and a directed graph, respectively. By (V, B), we denote a graph with vertex set V and arc set B. For two distinct vertices u and w, a u-w path is a path which begins at u and ends at w.

We consider a model of directed communication nets, called a file transmission net $N = (V, B, c_V, d, c_f)$, simply denoted by N, where (V, B) is the structure of N such that with each vertex v, two positive parameters $c_V(v)$ and $d(v)$ are associated. These parameters $c_V(v)$ and $d(v)$ are called the copying cost and the copy demand number, respectively, of v. For an arc e, $c_f(e)$ is the cost of transmitting copies through e, which is defined below. In this report, suppose that N has a path graph structure such as $V = \{v_i \mid i=1,2,\cdots,n\}$ and $B = \{(v_i, v_{i+1}) \mid i=1,2,\cdots,n-1\}$. Let denote an arc (v_i, v_{i+1}) by e_i. For a vertex v_j, let $An(v_j) = \{v_i \mid 1 \leqq i < j\}$ and $De(v_j) = \{v_i \mid j < i \leqq n\}$. A vertex v such that $An(v) = \phi$ is called a root and a vertex w such that $De(w) = \phi$ is called a leaf. In such N, v_1 is the root and v_n is the leaf. For simplicity, for a vertex v, let two vertices v' and v" satisfy $(v', v) \in B$ and $(v, v") \in B$, respectively. Note here that v_1' and v_n" are undefined.

Suppose that (1) some information to be distributed through N has been written in a file, (2) the written file is denoted by J, where the file means an abstract concept of information carrier and (3) to the root v_1, the original file of J is given from the outside of N. Suppose that on any vertex in V, we can duplicate J and we do not matter the difference from copies of J. In this situation, $c_V(v)$ means the cost of making a copy of J on a vertex v, and $d(v)$ means the number of copies of J needed on v. For $U \subseteq V$, let $d(U) = \Sigma_{u \in U} d(u)$.

The original file of J given to v_1 is duplicated and distributed to vertices v in such a way that $d(v)$ copies of J are taken out from v to the outside of N based on a file transfer defined below.

Definiton1: In N, if a function Ψ such that $\Psi : V \rightarrow Z_0^+$ and a function f such that $f : B \rightarrow Z^+$ satisfies

$$\begin{aligned} &\Psi(v_1) = f(e_1) + d(v_1), \\ &f(e_{i-1}) + \Psi(v_i) = f(e_i) + d(v_i) \ (1 < i < n) \quad (C1), \\ &f(e_{n-1}) + \Psi(v_n) = d(v_n), \end{aligned}$$

then (Ψ, f) is called a file transfer on N. For a function f on B, let c_f satisfy

$$c_f(e) = \begin{cases} 0 & (f(e)=0), \\ C_1 & (0 < f(e) \leqq \ell), \\ C_2 & (f(e) > \ell), \end{cases}$$

where ℓ, C_1, $C_2 \in Z^+$ and $C_1 < C_2$. For a file transfer $D = (\Psi, f)$ on N, let

$$C(D) = \Sigma_{v \in V} c_V(v) \Psi(v) + \Sigma_{e \in B} c_f(e), \quad (1)$$

which is called the cost of D. A file transfer with the minimum cost is said to be optimal. □

In the following, unless otherwise stated, file transfer is always defined to be on N. In a file transfer (Ψ, f), $\Psi(v)$ is the number of copies of J made at the vertex v and f (e) is the number of copies of J transmitted along the arc e. Our file transfer problem is defined as follows.

[File Transfer Problem] For a given file transmission net N, how do we obtain an optimal file transfer ? □

This file transfer problem is NP-hard even when N contains only two vertices with parallel arcs [11]. Therefore, we restrict N to solve the problem in polynomial time. As mentioned above, we assume that N has a directed path graph structure without parallel arcs. In [13], we suppose that $d(v)=1$ for each $v \in V$, which we sometimes call *simple demand* case, and propose $O(n^2)$ algorithm to solve the file transfer problem. In this report, we suppose that each vertex demand is positive integer. To make distinction, we call this case *plural demand* case.

If $|V| = 1$, then we have a unique file transfer which is trivially optimal. Therefore, in the following, $|V| \geqq 2$. For each file transfer, we define relations between vertices as follows.

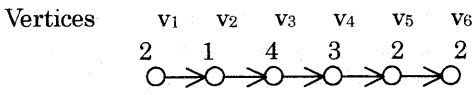

Fig.2-1. A file transmission net N

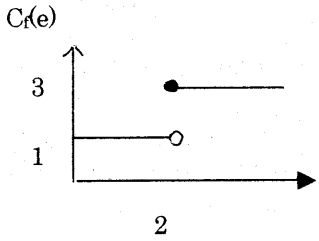

Fig.2-2. The cost of each arc e on N

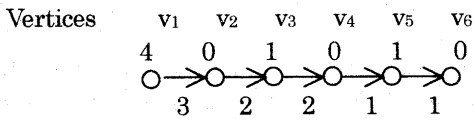

Fig.2-3. A file transfer (Ψ, f)

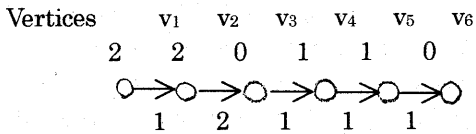

Fig.2-4. An optimal file transfer (Ψ', f')

Definiton2: In a file transfer (Ψ,f), we say that copies are made at a vertex v or v is a copying vertex if $\Psi(v)>0$. The copying vertex set in (Ψ,f) is $\{v\in V|\Psi(v)>0\}$. We say that the copying vertex set $W=\{w_i|1\leqq i\leqq s\}$ is in this order if W satisfies $w_{j+1} \in De(w_j)$, with $1\leqq j < s$. If w is a copying vertex and for a vertex set U, each vertex $u\in De(w)\cap U$ satisfies $\Psi(u)=0$, then we say that w is the last copying vertex in U, or copies are made last at w in U. If $U=V$, then we just call such w the last copying vertex. If w is a copying vertex and for a vertex set U, each vertex $u\in An(w)\cap U$ satisfies $\Psi(u)=0$, then we say that w is the first copying vertex in U. If both x and $y\in De(x)$ are copying vertices and x-y path contains no copying vertex but x and y, then we say that y is the next copying vertex to x or x is the previous copying vertex to y. □

In the following figures, unless otherwise stated, a file transfer (ϕ, f) indicates the values near a vertex v and an arc e denote $\phi(v)$ and f(e), respectively.

Here are examples of the above definitions. A file transmission net N with simple demand case is shown in Fig.2-1 and Fig.2-2. In Fig.2-1, the value near a vertex indicates its copying cost. We have a file transfer (ϕ ,f) shown in Fig.1-3, with its cost of $2\times4+4\times1+2\times1+3\times1+1\times4=21$. We have an optimal file transfer (ϕ', f') with its cost 16 shown in Fig.2-4.

3. Algorithm

In this section, we propose an algorithm to find an optimal file transfer. First, some necessary definitions are made.

Definiton3: Let (V_k, B_k) denote the structure of a subnetwork of N with a vertex set $V_k=\{v_1,v_2,\cdots,v_k\}$ and an arc set $B_k=\{e_1,e_2,\cdots,e_{k-1}\}$. □

Note that N_1 is composed of one vertex with $B_1=\phi$. Based on vertex demands, we define the following subnetwork.

Definiton4: Let $k\in Z^+$. Then the subnetwork $N_k=(V_{k'}, B_{k'}, c_V', d', c_f')$ is with the structure of $(V_{k'}, B_{k'})$ for $k'\in Z^+$, satisfying $d(V_{k'-1})< k \leqq d(V_{k'})$,

$$c_V'(v)=c_V(v) \quad (v\in V_{k'}),$$
$$d'(v)=d(v) \quad (v\in V_{k'-1}),$$
$$d'(v_{k'})=k-d(V_{k'-1}),$$
$$c_f'(e)=c_f(e) \quad (e\in B_{k'}).$$ □

Note here that $d'(V_{k'})=k$ and that the structure of N_{k+1} is different from that of N_k for some k'.

In the following, a net of N, denoted by $N(\alpha,\beta)$, means a network whose structure is identical to that of N and with each vertex v and each arc e, $\alpha(v)$ and $\beta(e)$ are associated, respectively. Note that file transfer itself is a net of

N.

Definiton5: Let $V_{k'}$ and $V_{k''}$ be the vertex set of N_k and N_{k+1}, respectively. Let $v_j \in V_{k'}$. For a net N_k (Ψ, f), functions Ψ' on $V_{k''}$ and f' on $B_{k''}$ are defined to be

$$\Psi'(v_j) = \Psi(v_j)+1,$$
$$\Psi'(v) = \Psi(v) \quad \text{(otherwise)},$$
$$f'(e_i) = f(e_i)+1 \quad (j \leqq i < k'),$$
$$f'(e) = f(e) \quad \text{(otherwise)}.$$

if k"=k' and otherwise, that is, k"=k'+1,

$$\Psi'(v_j) = \Psi(v_j)+1,$$
$$\Psi'(v_{k'+1}) = 0,$$
$$\Psi'(v) = \Psi(v) \quad \text{(otherwise)},$$
$$f'(e_i) = f(e_i)+1 \quad (j \leqq i < k'),$$
$$f'(e_{k'}) = 1,$$
$$f'(e) = f(e) \quad \text{(otherwise)}.$$

The operation to obtain the net $N_{k+1}(\Psi', f')$ is called path increment from v_j or $v_j - v_{k''}$ path increment for N_k (Ψ, f). □

Obviously, path increment for a file transfer on N_k yields a file transfer on N_{k+1}. Note here that it might happen that $v_j = v_{k'}$, $j \neq k$, that is, just increasing Ψ of the leaf by one, in the former case.

3.1. Algorithm for simple demand case[13]

Definition6: Let v_j and v_k be vertices such that $v_j \in An(v_k)$. For a file transfer $D=(\Psi, f)$, functions Ψ' on V and f' on B are defined to be

$$\Psi'(v_j) = \Psi(v_j) - 1,$$
$$\Psi'(v_k) = \Psi(v_k)+1,$$
$$\Psi'(v) = \Psi(v) \quad \text{(otherwise)},$$
$$f'(e_i) = f(e_i) - 1 \quad (j \leqq i < k),$$
$$f'(e) = f(e) \quad \text{(otherwise)}.$$

The operation to obtain the net $N(\Psi', f')$ from D is called a copy exchange from v_j to v_k for D with one copy. □

Definiton7: Let v_j and v_k be vertices such that $v_j \in An(v_k)$. For a file transfer $D=(\Psi, f)$, functions Ψ'' on V and f'' on B are defined to be

$$\Psi''(v_j) = \Psi(v_j) + z,$$
$$\Psi''(v_k) = \Psi(v_k) - z,$$
$$\Psi''(v) = \Psi(v) \quad \text{(otherwise)},$$
$$f''(e_i) = f(e_i) + z \quad (j \leqq i < k),$$
$$f''(e) = f(e) \quad \text{(otherwise)}.$$

where $z \in Z^+$. The operation to obtain the net N (Ψ'', f'') from D is called a copy exchange from v_k to v_j for D with z copies for D. □

The following definition is a key to consider optimalting

Definiton8: Suppose that a copying vertex set W satisfies $|W| \geqq 2$ in a file transfer $D=(\Psi, f)$. The operation to obtain another file transfer by a copy exchange from the last copying vertex w to its previous copying vertex with $\Psi(w)$ copies is called a backward operation for D. The process of continuing possibly backward operations and obtain a file transfer with the minimum cost among $|W|$ file transfers is called *full* backward operation for D. We identify those obtained file transfers with the use of the last copying vertex, say x, by denoting $D[x]$ and call a backward operation till x for D. □

In [13], with the use of forward and backward operations, we propose $O(n^2)$ algorithm as follows.

[Algorithm]
(Input) A file transmission net N with simple demand vertex
(Output) An optimal file transfer (Ψ_n', f_n')
Step1. Let $\Psi_i'(v_i) \leftarrow -1$.
Step2. For each integer i, with i=1,2,⋯, n−1, perform the following operations.
2-1. Perform Forward operation for a file transfer (Ψ_i', f_i') on N_i and obtain a file transfer D_{i+1} on N_{i+1}.
2-2. Perform *full* backward operation for D_{i+1} and obtain a file transfer (Ψ_{i+1}', f_{i+1}') on N_{i+1}. (Termination)

[Forward operation]

(Input) A file transfer (Ψ_i', f_i') on N_i.

(Output) A file transfer (Ψ_{i+1}, f_{i+1}) on N_{i+1}.

Step1. Let w be the last copying vertex in (Ψ_i', f_i').

Step2. Obtain (Ψ_{i+1}, f_{i+1}) as follows.

2-1. If $c_V(w) > c_V(v_i)$, then let $w \leftarrow v_i$.

2-2. Otherwise if $De(w) \cup \{w\}$ has a vertex x such that $f_i'(x, x'') = \ell$, then choose a vertex u on x''-v_i path with the minimum copying cost. In addition, if $c_V(u) < c_V(w) + C_2 - C_1$, then let $w \leftarrow u$.

2-3. Make $w - v_{i+1}$ path increment for (Ψ_i', f_i'). (Termination)

3.2. Plural demand case

Similarly to the simple demand case, it is easy to see that we can get an optimal file transfer on each subnetwork by one forward operation and one backward operation. Therefore, we skip here to show the proof of its optimality and refer to that in [13]. However, in the similar way to the simple demand case, the algorithm for plural case becomes $O(|V| d(V))$. So from now on, we consider how to reduce the times of backward operations and show the total time complexity of the algorithm for plural demand case is $O(n^2)$.

Before going to the revised algorithm, we observe some differences between simple and plural demand cases.

First, in the plural demand case, we have to consider the leaf itself as a possible copying vertex if its demand is more than one, while in the simple demand case we do not need to consider that.

Second, in the simple demand case, if the last copying vertex w satisfies $f(w,w'') \geqq \ell$, then w''- v_{i+1} path contains just one vertex such that $f(x, x'') = \ell$. However, in the plural demand case, if $De(w)$ contains a vertex v with $d(v) \geqq 2$, then it might happen that $f(v',u) \geqq \ell+1$ and $f(v, v'') \leqq \ell - 1$, that is, w''- v_{i+1} path contains no vertex such that $f(x, x'') = \ell$.

Third, in simple demand case, if backward operation till a vertex x was performed, then every copying vertex in $De(x)$ never becomes a copying vertex again. However, in plural demand case, it might happen that some "old" copying vertex becomes a copying vertex again. Here's an example. In Fig.3-1, we have a file transmission net N_{10} with $C_2 - C_1 = 5$ and $\ell = 5$. To obtain an optimal file transfer on that net, we consider optimal ones on its subnetworks N_8 and N_9. Actually both nets have unique optimal file transfers (Ψ_8, f_8) and (Ψ_9, f_9) shown in Fig.3-2 and 3-3, respectively. While, N_{10} has an optimal file transfer, which is unique too, shown in Fig.3-4. It is easy to see that v_3 is a copying vertex in (Ψ_8, f_8) and (Ψ_{10}, f_{10}), but not in (Ψ_9, f_9).

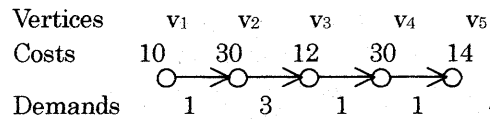

Fig.3-1. A file transmission net N_{10}

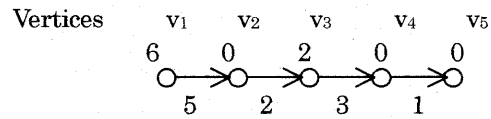

Fig.3-2. A file transfer (Ψ_8, f_8) on N_8

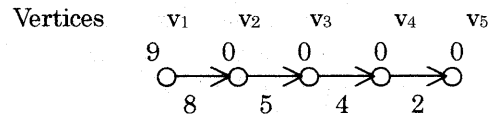

Fig.3-3. A file transfer (Ψ_9, f_9) on N_9

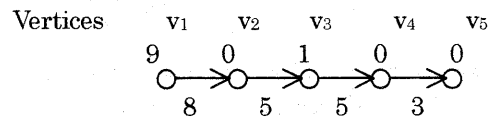

Fig.3-4. An optimal file transfer (Ψ_{10}, f_{10}) on N_{10}

Here, we propose the following algorithm for plural demand case.

[Revised Algorithm]

(Input) A file transmission net N with plural demand vertex

(Output) An optimal file transfer (Ψ, f)

Step1. For j=1,2,⋯,n, let $j_i = d(V_i)$. Let $\Psi(v_1) \leftarrow d(v_1)$ and $w \leftarrow v_1$.

Step2. For each integer j, with j= 2,3,⋯,n, perform the following operations.

2-1. Let $cnt \leftarrow d(v_j)$.

2-1-1. If $c_V(w) > c_V(v_{j-1})$, then $w \leftarrow v_{j-1}$.

2-1-2. Otherwise if $D_e(w) \cup \{w\}$ has a vertex x such that $f(x', x) = \ell$, then choose a vertex u on $x''-v_{j-1}$ path with the minimum copying cost. In addition, if $c_V(u) < c_V(w)+C_2-C_1$, then let $w \leftarrow u$.

2-1-3. Make $w-v_j$ path increment, $cnt \leftarrow cnt-1$, and perform full backward operation.

2-2. While(cnt > 0) repeat the following.

2-2-1. If $c_V(w) > c_V(v_j)$, then $w \leftarrow v_j$.

2-2-2. Otherwise if $D_e(w) \cup \{w\}$ has a vertex x such that $f(x', x) = \ell$, then choose a vertex u on $x-v_j$ path with the minimum copying cost. In addition, if $c_V(u) < c_V(w)+C_2-C_1$, then let $w \leftarrow u$.

2-2-3. If $f(w', w)+cnt \leqq \ell$, then make $w-v_j$ path increment cnt times and perform full backward operation, return to Step2-1. Otherwise, make $w-v_j$ path increment $\ell - f(w', w)$ times, $cnt \leftarrow cnt + f(w', w) - \ell$ times, perform full backward operation and return to Step 2-2. (Termination)

Before analyzing the new algorithm, we consider the validity of reducing backward operations.

Lemma1: In the algorithm, let D [x] be a file transfer obtained by backward operation from another file transfer D. Then every copying vertex w in $D_e(x)$ on D satisfies $c_V(w) \geqq c_V(x)$.

(Proof) We can get another file transfer by copy exchange from x to w. Since D[x] is optimal, either $-c_V(x)-C_2+C_1+c_V(w) \geqq 0$ or $-c_V(x)+c_V(w) \geqq 0$ holds. □

Lemma2: In Step2-2 of the algorithm, if another file transfer D [x] on N_j is obtained by backward operation for D, then the copying vertex w in $D_e(x)$ on D becomes a copying vertex by the next path increment for D [x] only if $f(w,w") < \ell$, $c_V(w) < c_V(x)+ C_2-C_1$, and x-w path contains an arc e such that $f(e)=\ell$.

(Proof) Lemma 1 says that all copying vertices w in $D_e(x)$ on D satisfies $c_V(w) \geqq c_V(x)$. So such vertex w becomes again a copying vertex if w has the above condition. □

Actually, the file transfers in Figs. 3-3 and 3-4 are the example of this lemma. From these lammas, the next proposition trivially holds.

Proposition1: Let v_j be the leaf of $N_{k'}$ to $N_{k"}$, with $k' > d(V_{j-1})$ and $k"=d(V_j)$. Suppose that we perform backward operation to obtain a file transfer $D_k[x]=(\phi_k, f_k)$ on N_k, with $k' < k \leqq k"$ from another file transfer D_k. Then no copying vertex in $D_e(x)$ on D_k as forward operations to obtain optimal file transfer except that $f(w,w") < \ell$, $c_V(w) < c_V(x)+ C_2-C_1$, and x-w path contains an arc e such that $f(e)=\ell$. □

The following lemmas are relevant to the analysis of the algorithm.

Lemma3: In the above algorithm, each arc e changes from $f(e)<\ell$ to $f'(e)=\ell$ at most one time, where the function f ' is obtained from f by path increment or backward operation.

(Proof) Because the above algorithm makes f non-decreasing. □

Lemma4: Backward operation is made at most 3 $|V|$ times in the above algorithm.

(Proof) We make that operation at Step 2-1 and 2-2. Other than that, when possible backward operation would happen for the case that $f(e)=\ell$. From the Lemma3, each arc becomes that change at most one time. □

Lemma5: In the algorithm, the pair of (w,v) such that w-v path increment is made in the algorithm is

at most 8 | V | times.

(Proof) Unless any backward operation is done, then the next copying vertex to w is always in De(w). So the total number of such pair is at most 2 | V | . In additon, if one backward operation is done, then only vertex x satisfying f(x', x)=ℓ can become a copying vertex again just after the operation. So by Lemma4, the total number of the pair (w,v) such that w-v path increment is made involved with backward operation is 2×3 | V | . □

Finally, we give a proposition about the time complexity of the above algorithm.

Proposition2: It takes $O(n^2)$ to implement the above algorithm.

(Proof) Lemma5 says that the total number of all path increments is $O(n)$. One path increment needs $O(n)$ operation in the algorithm. □

We mention that the space complexity is O (n), though at least d (V) different file transfers do appear in the algorithm.

4. Conclusion

In this report, we have considered the file transfer problem for a file transmission net with a directed path graph structure such that each arc cost is a two-level step and each vertex copy demand is positive. As a result, we have proposed an $O(n^2)$ algorithm to solve the problem, where n is the number of vertices.

References

[1] S.L.Hakimi, "Optimum locations of switching centers and the absolute centers and medians of a graph," Ops. Res. 12, 450-459 (1964).

[2] H.Tamura, M.Sengoku, S.Shinoda & T.Abe, "Location problems on undirected flow networks, IEICE Trans., E73, 12, pp.1989-1993, Dec.1990.

[3] H.Choi, S.Hakimi, "Data transfers in networks with transceivers, Networks 17, 393-421 (1987).

[4] I.Rivera-Vega, R.Varadarajan, & S.B.Navathe, " Scheduling file transfers in fullyconnected networks," Networks, 22, 563-588 (1992).

[5] R.Casey, "Allocation of copies of file in an information network," AFIPS Conf., Proc, 1972 Spring Joint Comput. Conf 40, 617-625 (1972).

[6] A.Segall, " Dynamic file assignment in a computer network," IEEE Trans. Automat. Contr. 20, 161-173 (1976).

[7] Y.Kaneko & S.Shinoda, "The complexity of an optimal file transfer problem," IEICE Trans. E82-A, 394-397, Feb.1999.

[8] Y.Kaneko, S.Shinoda & K.Horiuchi, " A synthesis of an optimal file transfer on a file transmission net," IEICE Trans. E76-A, 377-386, Mar.1993.

[9] Y.Kaneko, K.Suzuki, S.Shinoda & K.Horiuchi, " "A synthesis of a forest-type optimal file transfer on a file transmission net with source vertices," IEICE Trans. E78-A, 671-679, June 1995.

[10] Y.Kaneko & S.Shinoda, "On the optimality of forest-type file transfers on a file transmission net," IEICE Trans. E83-A, 1411-1419, July 2000.

[11] Y.Kaneko & Y.Shichiri, "On synthesizing a file economical scheduling on a file transmission net with step arc cost," Congressus Numerantium 114, 161-166, (1996).

[12] Balakrishnan & Ranganathan: A Textbook of Graph Theory, Springer (2000).

[13] Y.Kaneko, "An optimal file transfer on a directed path network with step arc costs," Congressus Numerantium 146, 173-186, (2000).

[14] M.R.Garey & D.S.Johnson: Computers and Intractability, A Guide to the Theory of NP-Completeness,W.H.Freeman, New York, (1979).