'87 KACC 1987.10.16～17

ON OVERLAPPING TERRITORIES SATISFYING CARDINALITY CONSTRAINTS

Takashi MORIIZUMI+, Shuji TSUKIYAMA+, Shoji SHINODA+
Masakazu SENGOKU*, and Isao SHIRAKAWA++

+ Dept. Electrical Eng.
Chuo University
Tokyo, 112 Japan

* Dept. Information Eng.
Niigata University
Niigata, 950-21 Japan

++ Dept. Electronic Eng.
Osaka University
Suita, 565 Japan

Abstract: Given a network with k specified vertices b_i called centers, a cardinality constrained cover is a family $\{B_i\}$ of k subsets covering the vertex set of a network, such that each subset B_i corresponds to and contains center b_i, and satisfies a given cardinality constraint. A set of cardinality constrained overlapping territories is a cardinality constrained cover such that the total sum of $T(B_i)$ for all subsets is minimum among all cardinality constrained covers, where $T(B_i)$ is the summation of the shortest path lengths from center b_i to every vertex in B_i.

This paper considers a problem of finding a set of cardinality constrained overlapping territories, and proposes an algorithm for the problem which has the time and space complexities are $O(k^3|V|^2)$ and $O(k|V|+|E|)$, respectively, where V and E are the sets of vertices and edges of a given network, respectively. The concept of overlapping territories has a possibility to be applied to a job assignment problem.

1. INTRODUCTION

Let N = (G,l) be a network such that G is a undirected connected graph with vertex set V and edge set E, and l is a function assigning a nonnegative real number l(e), called **edge length**, to each edge e∈E. In the network, k vertices b_1, $b_2, \ldots, b_k$ are specified, which are called **centers** or generators. For each center b_i, B_i denotes a set of distinct vertices containing b_i, and if family BB = { B_i } of these sets B_i ($1 \leqq i \leqq k$) satisfies the equality $\cup_{i=1}^{k} B_i = V$, that is, the union of subsets B_i of V is equal to V, then BB is called a **cover** of V. In the following, we denotes a cover BB = { B_i | $1 \leqq i \leqq k$ } simply by BB = { B_i }.

In a given network N, the **distance** d(u,v) from vertex u to vertex v is the shortest path length from u to v, where a length of a path is the total sum of edge lengths in the path. For subset B_i of a center b_i, let us define

$$T(B_i) = \Sigma_{v \in B_i} d(b_i, v),$$

and for a cover BB = { B_i }, let

$$T(BB) = \Sigma_{B_i \in BB} T(B_i).$$

Then, a cover BB with the minimum T(BB) among all covers is called a **set of overlapping territories**, and each subset B_i of BB is called a **territory** of b_i. We can easily see that if there is no constraint on the cardinality of each subset B_i, then overlapping territories BB is a (ordinary) territories, that is, each territory is pairwise disjoint and the cover BB is an exact cover or a partition of V.

A cover BB is called a **cardinality constrained cover** (abbreviated simply **CC cover**), if the cardinality |Bi| of each subset Bi of BB satisfies a given condition $M_i \leqq |B_i| \leqq N_i$ for the subset specified by two integers M_i and N_i such that

$$\Sigma_{i=1}^{k} M_i \leqq |V| \leqq \Sigma_{i=1}^{k} N_i.$$

A CC cover BB with the minimum value T(BB) among all CC covers is designated as a **set of cardinality constrained overlapping territories** (abbreviated simply **CCOT**).

In this paper, we consider a problem of finding a CCOT under the assumption that two integers Qj and Rj are assigned to each vertex v_j such that vertex v_j must belong to Qj different subsets and may not belong to more than Rj different subsets, where these Qj and Rj satisfy the following conditions:

$$1 \leqq Qj \leqq Rj \leqq k, \text{ for } 1 \leqq j \leqq k,$$
$$\Sigma_{i=1}^{k} M_i \leqq \Sigma_{v_j \in V} Rj,$$
$$\Sigma_{v_j \in V} Qj \leqq \Sigma_{i=1}^{k} N_i.$$

Hence, if Qj = Rj = 1 for every vertex v_j, then a desired CCOT becomes a set of cardinality constrained (ordinary) territories. Moreover, if there is no cardinality constraint on every subset B_i, then territories can be obtained from a Voronoi diagram[1,2] on network N by putting every vertex with more than one nearest centers into one of the Voronoi polygons of the nearest centers.

The concept of CCOT has a possibility to be applied to a scheduling problem, especially a job assignment problem. For example, each project (or job) corresponds to a center (representing project leader) and has two integers M and N such that at least M and at most N persons (or processors) are necessary to execute the job. Each employee (or processor) corresponds to a vertex and has an integer R representing the number of jobs which can be performed simultaneously. Graph G of network N = (G,l) denotes connection between persons (or processors) and edge length indicates distance or strength of the connection. In such a model, CCOT gives an optimal job assignment such that all the jobs are executed in a short time.

In the following, we use basic terminologies of a graph and a network without definitions (See [3,4] for definitions).

2. ASSOCIATED GRAPH AND UPDATE PROCESS

Given a CC cover BB = { B_i } for network N = (G,l), let us define a value $l*(b_i,v_j)$ for center b_j and vertex v_j in subset B_j and center b_i in subset B_i as follows;

$$l*(b_i,v_j) = d(b_i,v_j) - d(b_j,v_j).$$

Then, this value represents the increment of T(BB) caused by displacing v_j from B_j to B_i. Namely, let BB' be a CC cover obtained from BB by displacing v_j from B_j to B_i, and we have

$$T(BB') = T(BB) + l*(b_i,v_j).$$

With the use of this value, we define a weighted digraph G*(BB) for a CC cover BB, which is called the **associated graph** of BB, as follows:

(i) Each vertex b_i* of G*(BB) corresponds to a subset B_i of BB.

(ii) For every ordered pair (b_i*,b_j*) of distinct vertices such that $B_j - B_i - \{b_j\}$ contains at least one vertex v_j, there exists a directed edge incident from b_i* to b_j* with the length $l*(b_i*,b_j*)$ such that

$$l*(b_i*,b_j*)= \min\{ l*(b_i,v_j) \mid v_j \in B_j-B_i-\{b_j\} \}.$$

For an edge $e* = (b_i*,b_j*)$ of G*(BB), let head(e*) be a vertex v_j such that $l*(b_i*,b_j*) = l*(b_i,v_j)$, and we can easily see that there holds the following equality for a CC cover BB' obtained from BB by displacing head(e*) from B_j to B_i:

$$T(BB') = T(BB) + l*(b_i,\text{head}(e*)) = T(BB) + l*(b_i*,b_j*).$$

Let us conduct this displacement process repeatedly for every edge e* on an elementary dipath P* of G*(BB).

<UPDATE PROCESS OF A COVER>

Step-1: Let $b_{i_1}*, b_{i_2}*, \ldots, b_{i_r}*$ be vertices appeared in this order on an elementary dipath P*, and set q = r - 1.

Step-2: While q > 0, conduct step-3 with respect to edge $e_q* = (b_{i_q}*,b_{i_{q+1}}*)$ on P*.

Step-3: Displace head(e_q*) from $B_{i_{q+1}}$ to B_{i_q}, and return to step-2 by setting q = q - 1.

By BB[P*], we denote a cover obtained from cover BB by this UPDATE PROCESS with respect to a dipath P* of G*(BB). Then, subset B_{i_1} and B_{i_r} of BB[P*] = { B_i } has exactly by one vertex more and less than the corresponding subsets B_{i_1} and B_{i_r} of BB, respectively, and other subsets of BB[P*] have exactly the same number of vertices as those of BB. Moreover, there holds the following lemma.

[Lemma 1]

Let P* be an elementary path from $b_{i1}*$ to $b_{ir}*$ in G*(BB) which contains no cycle of negative length, and we have the following inequality;

$$T(BB[P*]) \leqq T(BB) + d*(P*),$$

where d*(P*) is the length of dipath P* in G*(BB). Moreover, if P* is a shortest path from $b_{i1}*$ to $b_{ir}*$, then the equality holds.

The proof of this lemma is omitted here.

This lemma indicates the following fact. Namely, let BB be a cover such that G*(BB) has no cycle of negative length, and let BB' be a cover obtained from BB by displacing a vertex v_j from B_j to B_i. Then, by applying UPDATE PROCESS stated above to BB with respect to shortest path P* from b_i* to b_j*, we can construct a cover BB[P*] such that T(.)

value is not greater than BB', that is,

$T(BB[P*]) = T(BB) + d*(P*) \leqq T(BB')$,

and every subset B_i of $BB[P*]$ has the same number of vertices as that of BB'.

Moreover, let $L*$ be an elementary cycle of $G*(BB)$, and we can regard $L*$ as a dipath starting from and ending at the same vertex. Therefore, if we apply UPDATE PROCESS to BB with respect to this $L*$, then we have a cover $BB[L*]$ such that every subset has the same number of vertices as that in BB and there holds

$T(BB[L*]) \leqq T(BB) + d*(L*)$,

where $d*(L*)$ is the length of cycle $L*$.

3. BASIC THEOREMS

In this section, we consider a method to modify an overlapping territories BB to a CCOT with the use of UPDATE PROCESS with respect to a shortest dipath in the associated graph $G*(BB)$. To do this, we have to define the following four sets $B_{mr}*$, $B_{ex}*$, $B_{ls}*$, and $B_{st}*$ of vertices of $G*(BB)$ with respect to a cover BB,

$B_{mr}* = \{ b_i* \mid M_i < |B_i| \}$,
$B_{ex}* = \{ b_i* \mid N_i < |B_i| \}$,
$B_{ls}* = \{ b_i* \mid |B_i| < N_i \}$,
$B_{st}* = \{ b_i* \mid |B_i| < M_i \}$,

where M_i and N_i are two specified integers representing constraints on the cardinality of subset $B_i \in BB$. From the definitions, we can readily see that $B_{mr}* \supset B_{ex}*$, $B_{st}* \subset B_{ls}*$, and BB is a CC cover if both $B_{ex}*$ and $B_{st}*$ are empty. Moreover, a set of territories BB is a desired CCOT, if both $B_{ex}*$ and $B_{st}*$ are empty and every v_j is contained in more than Qj-1 and less than Rj+1 subsets. We also see that subsets B_j and B_i corresponding to vertices $b_j* \in B_{mr}*$ and $b_i* \in B_{ls}*$, respectively, are the ones which satisfy given cardinality constraints even if the numbers of vertices in the subsets are decreased and increased by one, respectively.

[Theorem 1]

A given CC cover $BB = \{ B_i \mid M_i \leqq |B_i| \leqq N_i, 1 \leqq i \leqq k \}$ is a CCOT, if and only if the associated graph $G*(BB)$ of a CC cover BB contains neither a cycle $L*$ of negative length $d*(L*) < 0$ nor a dipath $P*$ of negative length $d*(P*) < 0$ from a vertex $b_i* \in B_{ls}*$ to a vertex $b_j* \in B_{mr}*$.

We omit the proof.

This theorem indicates a necessary and sufficient condition for a CC cover to be a CCOT, while the following theorems imply a method to modify a set of overlapping territories to a CCOT. We also omit the proofs of the theorems.

[Theorem 2a]

In the associated graph $G*(BB)$ of a CCOT $BB = \{ B_i \mid M_i \leqq |B_i| \leqq N_i, 1 \leqq i \leqq k \}$, let P_0* be a shortest dipath from a vertex b_s*, corresponding to subset B_s with $M_s = |B_s| \leqq N_s$, to a vertex $b_j* \in B_{mr}*$, and let $BB[P_0*] = BB' = \{B_i'\}$ be a cover obtained from BB by UPDATE PROCESS with respect to P_0*. Then, $BB[P_0*]$ is a CCOT such that subset B_s' corresponding to b_s* satisfies $M_s+1 = |B_s'| \leqq \max(N_s, |B_s|+1)$, and any other subset B_i' satisfies $M_i \leqq |B_i'| \leqq N_i$.

[Theorem 2b]

In the associated graph $G*(BB)$ of a CCOT $BB = \{ B_i \mid M_i \leqq |B_i| \leqq N_i, 1 \leqq i \leqq k \}$, let P_0* be a shortest dipath from a vertex $b_j* \in B_{ls}*$ to a vertex b_t*, corresponding to subset B_t with $M_t \leqq |B_t| = N_t$, and let $BB[P_0*] = BB' = \{B_i'\}$ be a cover obtained from BB by UPDATE PROCESS with respect to P_0*. Then, $BB[P_0*]$ is a CCOT such that subset B_t' corresponding b_t* satisfies $\min(M_s, |B_t|-1) \leqq |B_t'| = N_s-1$, and any other subset B_i' satisfies $M_i \leqq |B_i'| \leqq N_i$.

In order to devise an algorithm to find a CCOT by using these theorems, we have to introduce a dummy center b_{k+1} into network N such that b_{k+1} is connected with every vertex v_j (including centers) by edges (b_{k+1}, v_j) of zero length $l(b_{k+1}, v_j) = 0$, and the corresponding subset B_{k+1} has the cardinality constraint such that $M_{k+1} = N_{k+1} = 1$. This dummy center b_{k+1} and its subset B_{k+1} are used to handle the multiplicity of a vertex, in such a way that B_{k+1} contains b_{k+1} and vertices v_j each of which is contained in less than Qj (or Rj) subsets of $\{ B_i \mid 1 \leqq i \leqq k \}$, and B_{k+1} is reconstructed, every time a CCOT is updated. Henceforth, we denote by m_j the number of multiplicity of vertex v_j in subsets $B_1, B_2, \ldots, B_k$, that is, the number of subsets B_i $(1 \leqq i \leqq k)$ containing v_j.

The outline of the proposed algorithm is as follows:

Firstly, each subset B_i ($1 \leqq i \leqq k$) contains center b_i alone, and subset B_{k+1} contains b_{k+1} and vertices v_j such that $m_j < Qj$. Then, we can easily see that the cover $BB^0 = \{ B_i^0 \mid 1 \leqq i \leqq k+1 \}$ thus obtained is a CCOT such that each B_i^0 ($1 \leqq i \leqq k$) satisfies $1 = |B_i^0| \leqq N_i$ and subset B_{k+1}^0 satisfies $1 \leqq |B_{k+1}^0| \leqq |V|$. Moreover, let $B_{st}*^0$ and $B_{mr}*^0$ be the sets of vertices similarly defined to $B_{st}*$ and $B_{mr}*$ in $G*(BB^0)$, respectively, then $B_{st}*^0 \subset \{b_i* \mid 1 \leqq i \leqq k\}$ and $B_{mr}* = B_{ex}* = \{b_{k+1}\}$.

Secondly, UPDATE PROCESS is applied to this BB^0 with respect to a shortest dipath P_0* from a vertex $b_s* \in B_{st}*^0$ to $b_{k+1}*$, and we can see from Theorem 2a that cover $BB^1 = BB^0[P_0*]$ obtained by UPDATE PROCESS with respect to P_0* is a CCOT such that $B_s^1 \in BB^1$ corresponding to b_s* satisfies $2 = |B_s^1| \leqq N_s$ and any other subset $B_i^1 \in BB^1$ satisfies the same constraint as in BB^0. Then, we reconstruct B_{k+1} of BB^1 as a subset containing b_{k+1} and vertices such that $m_j < Qj$, since the vertex displaced from B_{k+1}^0 by UPDATE PROCESS may be contained in B_{k+1}^1.

Thirdly, the process described above are repeated until $B_{st}*$ becomes empty or $B_{k+1} = \{b_{k+1}\}$. If $\Sigma_{i=1}^{k} M_i > \Sigma_{v_j \in V} Qj$, then we have $B_{k+1}^h = \{b_{k+1}\}$ and $B_{st}*^h \neq \phi$ after a certain number, say h, of iteration; otherwise there holds $B_{st}*^h = \phi$, before B_{k+1}^h becomes $\{b_{k+1}\}$.

In the case of $B_{k+1}^h = \{b_{k+1}\}$, we have a CCOT BB^h such that every vertex v_j is contained in Qj different subsets, and each subset $B_i \in BB^h$ satisfies

$$M_i^h = |B_i^h| < N_i \ ; \text{ if } B_i^h \in BB_{st}^h,$$

$$M_i \leqq |B_i^h| \leqq N_i \ ; \text{ if } B_i^h \notin BB_{st}^h,$$

where BB_{st}^h is a family of subsets B_a corresponding to a vertex $b_a* \in B_{st}*^h$ and M_i^h is an integer equal to $|B_i^h|$. Thus, we reconstruct B_{k+1}^h of BB^h as a subset containing b_{k+1} and vertices v_j such that $m_j < Rj$, and repeat UPDATE PROCESS similarly to the above. Then, we finally obtain a desired CCOT when $B_{st}*$ becomes empty.

In the case when $B_{st}*^h$ becomes empty in the h-th iteration, if $B_{k+1}^h = \{b_{k+1}\}$ at that time, then we have a desired CCOT BB^h at the same time. Otherwise, we have to distribute the remaining vertices in B_{k+1}^h exclusive of b_{k+1} to appropriate subsets. Thus, by using Theorem 2b, we repeat UPDATE PROCESS with respect to a shortest dipath from a vertex of $B_{1s}*$ to $b_{k+1}*$, until B_{k+1} becomes a singleton. Then, we finally have a desired CCOT. In the repetition, we have to reconstruct B_{k+1} as a subset containing b_{k+1} and vertices v_j such that $m_j < Qj$, every time a cover is updated.

4. ALGORITHM

Before describing the details of the proposed algorithm, let us consider a method to find a shortest dipath in an associated graph $G*(BB)$.

Although a given network N has no edge of negative length, the associated graph $G*(BB)$ of a cover BB may have such an edge, and hence we cannot use Dijkstra-like algorithm[3,5] for finding a shortest path. However, in each iteration of the algorithm, the associated graph $G*(BB^h)$ is constructed for a certain CCOT BB^h, and hence we can see from Theorem 1 that $G*(BB^h)$ has no cycle of negative length. Therefore, as is well known in algorithms for the minimum-cost flow problem[3,5], we can modify the edge length $l*(e*)$ of each edge $e*=(b_u*,b_v*)$ in $G*(BB^h)$ to a nonnegative edge length $l**(e*)$ by the following equation;

$$l**(b_u*,b_v*) = l*(b_u*,b_v*) + p(b_u*) - p(b_v*),$$

where p is a function assigning to each vertex v_j a nonnegative real value $p(v_j)$ (usually the distance from the origin of the generated shortest path). By this transformation, the length $d*(L*)$ of a cycle $L*$ is not changed, and the length $d*(P*)$ of a dipath $P*$ from b_s* to b_t* is changed to $d**(P*) = d*(P*) + p(b_s*) - p(b_t*)$, and hence if $p(b_s*) = 0$, the original length $d*(P*)$ of dipath $P*$ is given by $d*(P*) = d**(P*) + p(b_t*)$.

The detailed description of the proposed algorithm is as follows:

<Initialization Step>

I-1: For every pair of center b_i and a vertex v_j, calculate the distance $d(b_i,v_j)$ from b_i to v_j, and store them in order to calculate $l*(b_i,v_j)$ easily.

I-2: Let $B_i = \{ b_i \}$ and $p(b_j*) = 0$ for $1 \leqq i \leqq k+1$. Then, $BB = \{ B_i \mid 1 \leqq i \leqq k+1 \}$ and $T(BB) = 0$.

I-3: Set flag = 0, which indicates what vertices are added to subset B_{k+1}.

<Iteration Step>

II-1: If flag = 0, then vertices v_j such that $m_j < Qj$ are added to B_{k+1}; otherwise (flag = 1), vertices v_j such that $m_j < Rj$ are added, where m_j is the number of subsets B_i ($1 \leqq i \leqq k$) containing vertex v_j.

II-2: If $B_{k+1} = \{b_{k+1}\}$ and $B_{st}* = \{ b_i* \mid |B_i| < M_i \} = \phi$, then BB - $\{ B_{k+1} \}$ is a desired CCOT, and hence terminate. If $B_{k+1} = \{b_{k+1}\}$ and $B_{st}* \neq \phi$, then set flag = 1 and return to II-1. If $B_{k+1} \neq \{b_{k+1}\}$ and $B_{st}* = \phi$, then go to III-1.

II-3: Construct the associated graph $G*(BB)$ of BB = $\{ B_i \mid 1 \leqq i \leqq k+1 \}$, and modify each edge length $l*(b_i*, b_j*)$ to

$$l**(b_i*, b_j*) = l*(b_i*, b_j*) + p(b_i*) - p(b_j*).$$

II-4: In network $(G*(BB), l**)$, calculate the shortest path length $D(b_i*)$ from a vertex in $B_{st}*$ to each vertex b_i*. Then, let P* be a shortest dipath from a vertex $b_s* \in B_{st}*$ to $b_{k+1}*$ whose length is $D(b_{k+1}*)$.

II-5: Apply UPDATE PROCESS to BB with respect to P*, and set

$T(BB) = T(BB) + D(b_{k+1}*) + p(b_{k+1}*)$.

II-6: Return to II-1, after setting $p(b_i*) = p(b_i*) + D(b_i*)$ for each $1 \leqq i \leqq k+1$ and $B_{k+1} = \{b_{k+1}\}$.

<Distribution Step>

III-1: Construct the associated graph $G*(BB)$ of BB and modify the edge length as described in II-3.

III-2: Introduce a new vertex b_0* and edges (b_0*, b_i*) of lengths $l**(b_0*, b_i*) = -p(b_i*)$ incident from b_0* to vertices b_i* of $B_{ls}* = \{ b_i* \mid |B_i| < N_i \}$.

III-3: Calculate the distance $D(b_i*)$ from vertex b_0* to every vertex b_i*, then let P* be a subpath from a vertex of $B_{ls}*$ to $b_{k+1}*$ of a dipath from b_0* to $b_{k+1}*$ whose length is $D(b_{k+1}*)$.

III-4: Apply UPDATE PROCESS to BB with respect to P*, and set

$T(BB) = T(BB) + p(b_{k+1}*) + D(b_{k+1}*)$.

Then, let B_{k+1} be a set containing b_{k+1} and vertex v_j such that $m_j < Qj$.

III-5: If $B_{k+1} = \{b_{k+1}\}$, then BB - $\{B_{k+1}\}$ is a desired CCOT, and hence terminate. Otherwise, return to III-1 after setting $p(b_i*) = p(b_i*) + D(b_i*)$ for each $1 \leqq i \leqq k+1$.

During <Iteration Step> in the algorithm, the number of vertices in a subset B_s corresponding to a vertex $b_s* \in B_{st}*$ increases one by one, and the number of vertices in $B_{st}*$ decreases gradually. Therefore, every time a shortest path from a vertex in $B_{st}*$ is calculated in II-4, there holds $p(b_s*) = 0$ for every $b_s* \in B_{st}*$. Thus, T(BB) value is correctly computed in II-5.

In III-2 of the algorithm, we introduced new edges of negative lengths. These edges and lengths are used for finding a shortest path from a vertex b_i* of $B_{ls}*$ to $b_{k+1}*$ correctly. Although these edge lengths are negative, we can use Dijkstra-like algorithm, since such edges are all restricted ones. Thus, we can prove the following lemma which is used for verifying the correctness of the algorithm, although we omit the proof.

[Lemma 2]

In the associated graph $G*(BB)$ constructed in any stage of the algorithm, we can calculate distance $D(.)$ correctly by Dijkstra-like algorithm.

Finally, we consider the time and space complexity of the algorithm.

It is not difficult to see that the space complexity is $O(k|V|+|E|)$, since $O(k|V|)$ space is necessary for storing distance $d(b_i, v_j)$ for every pair of center b_i and vertex v_j, $O(k^2)$ for an associated graph $G*(BB*)$, and $O(|V|+|E|)$ for all others, where V and E are the sets of vertices and edges of a given network, respectively.

As for the time complexity, <Initialization Step> from I-1 to I-3 require $O(k|V|^2)$ time, since the distances from a single source to all vertices can be calculated in $O(|V|^2)$ by Dijkstra-like shortest path algorithm. The construction of an associated graph in II-3 or III-1 requires $O(k^2|V|)$ time, since we have to check all pairs of subsets and each subset may have $O(|V|)$ vertices. The distance calculation in II-4 or III-3 can be executed in $O(k^2)$ time by Dijkstra-like algorithm, and all other processes can be done in $O(k)$ time. Therefore, one iteration of <Iteration Step> from II-1 through II-6 or <Distribution Step> from III-1 through III-5 requires $O(k^2|V|)$ time. On the other hand, the number of iterations of <Iteration step> or <Distribution Step>

is at most $k|V|$ times, since every subset B_i ($1 \leq i \leq k$) has at most $|V|$ vertices and exactly one vertex is added to a subset B_i ($1 \leq i \leq k$) in II-5 or III-4. Thus, we can see that the time complexity of the algorithm is $O(k^3|V|^2)$ totally.

5. CONCLUSION

In this paper , we have considered a problem of finding a set of overlapping territories which satisfies given cardinality constraint on every territory and multiplicity constraint on every vertex, and have proven a few theorems which imply an algorithm to find a set of cardinality constrained overlapping territories in $O(k^3|V|^2)$ time, where k and $|V|$ is the numbers of centers and vertices of a given network, respectively. Similar discussion can be done when each center and/or each vertex have weights, and T(BB) is defined with the use of these weights.

Concerning territories, there remains some problems to be considered, among which of primary importance is a problem of finding territories such that each territory B_i induces a connected subgraph of a given network.

REFERENCES

[1] F.P.Preparata, M.I.Shamos: Computational Geometry, Springer-Verlag, New York, pp.198-257, 1985.

[2] T.Asano, M.Sato, T.Ohtsuki: "Computational geometry algorithms," Layout Design and Verification, Ed. T.Ohtsuki, North-Holland, Tokyo, pp. 295-347, 1986.

[3] M.Iri, I.Shirakawa, Y.Kajitani, S.Shinoda, et al.: Graph Theory with Exercises, Corona Publishing Co., Ltd, Tokyo, 1983 (in Japanese).

[4] F.Harary: Graph Theory, Addison-Wesley, Reading, 1969.

[5] E.L.Lawler: Combinatorial Optimization: Networks and Matroids, Holt, Rinehart and Winston, New York, 1976.