Dokl. Akad. Nauk SSSR
Tom 189 (1969), No. 1

Soviet Math. Dokl.
Vol.10 (1969), No. 6

AN ALGORITHM FOR THE SOLUTION OF THE ASSIGNEMT PROBLEM

UDC 518.5

E. A. DINIC AND M. A. KRONROD

Let (a_{ij}) be a square matrix of order n. A solution of the assignment problem for this matrix is a set of n elements of the matrix, one in each row and each column, such that the sum of these elements is minimal with respect to all such sets. Algorithms are known which solve the assignment problem after Cn^4 operations,* for example the Bradford method.

In the present article an algorithm is constructed which solves the problem more rapidly: one modification after $Cn^3 \log n$ operations, the other after Cn^3 operations.

Definition. Let some vector $\Delta = (\Delta_1, \Delta_2, \cdots, \Delta_n)$ be given. An element a_{ij} is called Δ-*minimal*, if $a_{ij} - \Delta_j \leq a_{ik} - \Delta_k$ for all k.

Lemma. *For any Δ let there be given a set of n Δ-minimal elements $a_{1j(1)}, a_{2j(2)}, \cdots, a_{nj(n)}$, one from each row and column. Then this set is a solution of the assinment problem.*

Proof.

$$\sum_{i=1}^{n} a_{ij(i)} = \sum_{k=1}^{n} \Delta_k + \sum_{i=1}^{n} (a_{ij(i)} - \Delta_{j(i)}).$$

The first sum in the right member is constant for all sets, and the second is minimal.

For any vector Δ let us select in each row one of the Δ-minimal elements (we shall call these selected elements basics). The number of free columns (columns without basics) is called the deficiency of the set of basics. From the lemma it follows that the set of basics with deficiency zero is a solution of the problem.

An algorithm is derived below permitting iteration: for a given vector Δ and a set of basics with nonzero deficiency, find a new vector and a set of basics possessing a lesser deficiency.

Taking the zero vector as Δ and some set of basics for it, we can solve the problem by successive iteration.

Iteration. For any vector Δ let us have a set of basics $a_{1j(1)}, a_{2j(2)}, \cdots, a_{nj(n)}$ with deficiency $m \neq 0$. We single out some free column. Let its index be s_1. We increase Δ_{s_1} to the maximum δ such that all basics remain Δ-minimal elements (a method of finding δ is indicated below). We obtain that for some i $a_{ij(i)} - \Delta_{j(i)} = a_{is_1} - \Delta_{s_1}$, i.e. the element a_{is_1} remains Δ-minimal. We call it an alternative basic and single out a column with index $s_2 = j(i)$, containing a basic of this row (we now have two marked columns). We increase Δ_{s_1} and Δ_{s_2} to the maximal δ such that all basics remain Δ-minimal, we find a new alternative basic in one of the columns and single out a new column, and so on until we single out a column with two or more basics.

Now we shall construct a new set of basics. We observe that in each row there is not more than one alternative basic.

* Contemplated is a number of operations realizable on a computer under a program realizing the algorithm. C is a multiplier not depending on n.

We shall call a change of basic in a row the following operation: an alternative basic in this row is declared basic, and the old basic ceases to be basic. We shall produce a change of basic in a row in which the last alternative basic lies. In this connection, in the last singled out column the number of basics is decreased by 1. In the column where the new basic appears we take the old basic and effect a change of basics in the corresponding row, and so on until a basic appears in the column with index s_1. Thereby we have obtained the vector Δ we require and the set of basics for it with deficiency equal to $m-1$.

It is easy to see that each iteration of the described algorithm requires Cn^2 operations, not counting operations entering into the calculation of δ at each stage of the iteration. Inasmuch as the number of iterations does not exceed n, only the time of computation of δ is of interest to us.

We shall now indicate various methods of computing δ. Let us consider an arbitrary stage of some iteration.

Let S be the set of indices of the marked columns, and R the set of indices of those rows in which there are no alternative basics. Then we have $\delta = \min_{i\in R, k\in S}[(a_k - \Delta_k) - (a_{ij(i)} - \Delta_{j(i)})]$. If at each step of the iteration to compute δ directly by this formula as the minimum of a group of a length of order n^2, then in all the algorithm requires Cn^4 operations.

We can write $\delta = \min_{k\in S}(\min_{i\in R}[(a_{ik} - \Delta_k) - (a_{ij(i)} - \Delta_{j(i)})])$; this formula is used for the first modification of the algorithm which permits one rapidly to find $\min_{i\in R}[(a_{ik} - \Delta_k) - (a_{ji(i)} - \Delta_{j(i)})]$ and which requires $Cn^3\log n$ operations, or to write $\delta = \min_{i\in R}[\min'_{k\in S}(a_{ik} - \Delta_k) - (a_{ji(i)} - \Delta_{j(i)})]$; this formula is used for the second modification, which permits one quickly to compute $\min_{k\in S}(a_{ik}\Delta_k)$ and requires Cn^3 operationa.

First modification. We introduce a certain method of listing the ordering of the group of numbers without permutation of its terms, permitting quick reaction to cancellation (without re-enumeration) of its elements.

Let a group of n numbers be given. For listing of the ordering, a group of n boxes of information is required. In the box corresponding to some element of the group we place the indices to the preceding and succeeding (in the ordering sense) elements, i.e. their numbers.

Let it be required for us to delete a certain element of the group. From its information box we know the indices of its neighbors in the ordering. We change their information boxes, placing instead of the indicators to the deleted element the indicators to its opposite neighbor. It is obvious that the remaining group will be a list of its ordering in the sense indicated above.

For the initial ordering $Cn \log n$ operations are required, for the change associated with a deletion C operations are required.

We use this method for rapid finding of $\min_{i\in R}[(a_{ik} - \Delta_k) - (a_{ij(i)} - \Delta_{j(i)})]$ for each $k \in S$.

Let $b_{ik} = [(a_{ik} - \Delta_k) - (a_{ij(i)} - \Delta_{j(i)})]$. Each of the columns of (b_{ik}) we order and list this ordering according to columns of a table of $n \times n$ information boxes. We write out separately the row of minima by columns, equal to $\min_i b_{ik}$ at the beginning of the iteration.

In the process of execution of an iteration we must delete from the matrix B the rows in which alternative basics appear, in order that the minimum by columns of the remaining matrix will be the sought $\min_{i\in R} b_{ik}$. The corresponding change in the information table and the row of minima, and also the finding of δ according to the formula $\delta = \min_{k\in S}(\min_{i\in R} b_{ik})$ requires Cn operations at each step of the iteration. But because the composition of the information table for each iteration requires $Cn^3 \log n$ operations, the total course of the algorithm requires $Cn^3\log n$ operations.

Second modification. We have $\delta = \min_{i \in R} [\min_{k \in S} (a_{ik} - \Delta_k) - (a_{ij(i)} - \Delta_{j(i)})]$. Let us consider the vector $q = (q_1, q_2, \cdots, q_n)$, where $q_i = \min_{k \in S} (a_{ik} - \Delta_k)$. We have $\delta = \min_{i \in R} [q_i - (a_{ij(i)} - \Delta_{j(i)})]$.

From this formula it follows that for solution of the problem after Cn^3 operations it is sufficient to be able to calculate at each step the vector q after Cn operations. We shall show how to do this.

At the first step of the iteration the vector q coincides with the first earmarked column. At each succeeding step the vector q is obtained from the old vector q also from a new earmarked column according to the following formula: $q_i' = \min (a_{is_m} - \Delta_{s_m}; q_i - \delta)$, where s_m is the index of the earmarked column.

Moscow State University

Received 27/FEB/69

Translated by:
R. F. Rinehart