Elementary Investigation of Transportation Problems

Edit Schmidt

Budapest Tech, Tavaszmező utca 17, H-1084 Budapest, Hungary
schmidt.edit@kvk.bmf.hu

Abstract: For students learning the simplex method of linear programming it is a well-beloved occasion to solve the so-called transportation problem by the method of distribution. This method is simple to calculate and easy to follow. The simple way of solution suggests that its correctness may be proven by basic means. This paper has two main aims. One of them is to present the problem and to solve it by basic means. The other one is the analysis of the so-called array-bases defined for this reason. In case of a transportation problem m stores and n destinations are given, and the goods have to be taken from the stores to the destinations such that the cost of transporting has to be minimal. The unit costs of the transportation are given by an array. In the solution some routes (elements of the array) are chosen and the number of units to transport there is given. It will be proven that the routes for the optimal transportation compose a basis, and the solution is also achieved by those through the searches. (The basis of an $m \times n$ *array consists of* $m+n-1$ *elements such that they do not span a loop.) In the proof some characteristics of the bases are needed, for example that the number of them is finite. To prove this it is enough to give an easily calculated upper bound, the exact value is given in the appendix. As an extra result of this calculation some interesting formulas of combinatorics are also proven.*

Keywords: transportation problem, linear algebra, bases

1 Basis

When solving a transportation problem, a rectangular array of entries (elements) of size $m \times n$ (an $m \times n$ matrix) is used. In the course of setting up the initial program, we assign certain elements, then change some of the assigned elements by specified rules. The algorithm is a special, graphic version of the simplex method. We can thus find an explanation of the steps to be made by studying the simplex method. ([1], [2].) The rules, however, can be proved without the "simplex background". The purpose of the present paper is to show this elementary proof.

The statements in the first chapter do not even need the definition of transportation problem (they make sense in themselves), but they seem to be useful in the following chapters. First of all, let us define some concepts.

Definition 1.1: Two entries of an array are *row-adjacent [or column-adjacent]*, if they are in the same row [or column]. In both cases they are *adjacent*.
A sequence of entries is called a *chain*, if adjacent entries follow each other. A chain is *closed*, if its first and last entries are adjacent.

Definition 1.2: A *loop* is a closed chain such that
- it contains more than 2 entries;
- consecutive entries are row- and column adjacent in turn;
- there is no repetitive entry in it;
- there is no repetitive row or column in it.

Before the following statement we note that a closed chain can be specified by listing its elements in several ways, and any of its elements can be chosen as first. The lemma will be used in the present chapter.

Lemma 1.3: Suppose that a closed chain is given such that its first element occurs only once in the chain, the second element is row-adjacent [column-adjacent], the last element is column-adjacent [row-adjacent] to it. Now by omitting some other elements of the chain, it can be reduced to a loop.

Proof: Let us follow the next procedure. Go along the closed chain three times (always starting with the first element), and omit some elements every time according to rules I, II and III.
I) If there are more than two row-adjacent [column-adjacent] consecutive entries, remove the middle ones and take the last one right after the first.
II) If we return to a point where we have been before remove the piece in between. If we arrived to the repetitive entry from a row [column] for the first time and from a column [row] for the second time, omit the entry itself, and the row-adjacent [column-adjacent] entry that preceded it for the first time has to be followed by the row-adjacent [column-adjacent] one that comes after it for the second time. As the first element occurs only once in the chain it cannot happen to be removed.
III) If we return to a row [column] where we have been before, then the first entry of the first occurrence has to be followed by the second row-adjacent [column-adjacent] entry of the second occurrence. The piece of the chain in between has to be removed. (Applying rule II assures that these entries are different.)
Complying with the rules above assures that the remaining closed chain is a loop.

When solving a transportation problem by distribution, we assign some routes to the initial program, and perform the transportation along these routes. The respective entries of the matrix are called tied elements. Thereinafter we are going to formulate and prove statements concerning tied elements. Let $m \geq 2, n \geq 2$ be integers. There is no distinguished role of rows and columns as compared to each other, thus every rule is valid by interchanging *m* and *n*.

Definition 1.4: A *basis* of an *m*×*n* array is $m+n-1$ *tied elements* such that they do not span a loop.

Lemma 1.5: Every array contains a basis.

Proof: We prove by induction.
In an array of size 2×2, every 3 of the elements are suitable, obviously. If an array is of size *k*×*g*, an adequate choice contains $k+g-1$ tied elements. Insert a new row [a new column] somewhere, and attach an arbitrary element of the new row [column] to the tied elements. Now we have an array of size $(k+1)\times g$ [$k\times(g+1)$] with $k+g$ tied elements. These elements do not span a loop as the original elements had not spanned one, and the new element cannot be a part of any loop because it is not row-adjacent [column-adjacent] to any of the elements. Thus, by starting off from a 2×2 array, and inserting $m-2$ rows and $n-2$ columns in arbitrary order, we obtain an array of size *m*×*n* with $m+n-1$ tied elements.

Definition 1.6: The *critical number* of an *m*×*n* array is $q=m+n-1$, i.e. the number of tied elements needed for a basis.

The basic properties of bases will be declared in the next lemma; it will be referred to later in this paper.

Lemma 1.7: An *m*×*n* array and one of its bases are given. They have the following properties:
1) There exists such a row or column which contains exactly one tied element, i.e. there is a “one-element” row or column; and there is no row and column without any tied element, i.e. there is no “empty” row or column.
2) A chain along tied elements can be drawn between any two elements of the basis.

Proof: 1) We prove first that there is such a row [column] which contains at most one tied element. Sum up the tied elements in every row and column. As any element occurs exactly in one row and in one column, the total will be $2(m+n-1)=2m+2n-2$. If every row and column contained at least two tied elements, the total would be greater than or equal to $(m+n)\cdot 2=2m+2n$. Thus there are (at least two) rows or columns which contain at most one tied element. Now we go on with proving the statement itself. In case of arrays of smaller size (e.g. of sizes 2×2, 2×3, i.e. if $q\leq 4$) it can be easily seen that it is true for the rows and columns of any basis: there is a one-element row or column, but there is no empty row or column among them. Suppose indirectly that the theorem is not always valid. Take a counter-example such that *q* has the smallest possible value, i.e. for any array that has a critical number less than this, the theorem is valid. Three cases have to be investigated. It cannot happen that the counter-example does not contain any row or column with less than two elements: it was shown at the beginning of the proof. If among the rows and columns of the counter-example there is an empty one, as well as one that contains exactly one element, remove the latter (thus the critical number decreases by 1). The remaining elements form a basis for the remaining array, as they do not contain a loop and they are of a sufficient number. This is a contradiction, as we have constructed a basis containing an empty row in an array of such size where it is not possible according to our assumption. If among the rows and columns of the counter-example there is not any that contains one element, but there is an empty one, then there must be two empty rows or columns, according to the beginning of the proof. Now remove an empty row or column and an arbitrary element. Now we have got to a contradiction again as at least one empty row or column remains. It is important to note that in case of a counter-example of two rows [two columns] the imagined removal does not lead to an array of one row [one column] (we do not deal with these cases thus the indirect assumption does not apply to them). This can be explained by the fact that an array of two rows [two columns] must contain a column [row] with exactly one tied element otherwise there would be a 4-element loop, as $q \geq 4$.

2) We show first that every tied element is adjacent to at least one element. Suppose indirectly that there exists a tied element with no adjacent element. Now remove its row [or column, as you like]. The remaining tied elements form a basis for the remaining array, still, it has an empty column [row], namely the column [row] of the omitted element. This contradicts property 1.

The proof is done by induction applied to the critical number of the array. In case of an array of size 2×2 and its 3-element bases (i.e. for $q = 3$), it is obvious that the tied elements form a chain. Suppose that in case of $q = r$ ($r \geq 3$) a chain can be drawn between any two tied elements of the basis. Now take an array and a (q-element) basis of it, for which $q = r + 1$. Remove temporarily one of its one-element rows [columns], such that the number of rows and columns be at least two. (This is possible, as shown above.) The remaining tied elements form a basis for the array left, thus according to the induction hypothesis a chain can be drawn between any two tied elements of the basis. The removed tied element did not have a row-adjacent [column-adjacent] element, thus as shown above, it must have had a column-adjacent [row-adjacent] element, that is part of the new basis. Now put back the removed row [column]. It only has to be proved that there is a chain between the removed (then returned) tied element and any other tied element. This chain can be found by stepping to the column-adjacent [row-adjacent] element from the removed (then returned) element, from there any further tied element can be reached along a chain, according to the above and to the induction hypothesis.

The question may arise how much a basis is characterized by these properties, e.g. whether they are definitive or not. Property 1 is not sufficient for q elements to form a basis. (It is easy to find a counter-example: take the elements with the following indices of a 3×3 array: $(1,1), (1,2), (2,1), (2,2), (3,3)$). On the other hand, it is not true either, that there cannot be found less than q elements such that this property holds. (Counter-example: take the elements with the following indices of a 3×3 array: $(1,1), (1,3), (2,2), (3,2))$.

Definition 1.8: In an array, a set of assigned elements is *dependent*, if a chain can be drawn between any two elements; it is *spanning* if every row and column contain a member of the set.

Theorem 1.9: If a set of q assigned elements of an $m \times n$ array is dependent and spanning, then these elements form a basis.

Proof: In an array of size 2×2 (i.e. in case of $q = 3$) any 3 of the elements form a basis, thus the existence of the two properties is sufficient. Suppose indirectly that the statement is not always valid. Take the counter-example where the value of q is minimal. In this case there is a loop along the assigned elements. Omit a one-element row [or column], the existence of which was shown at the beginning of the previous proof (as there is no empty row or column because of the spanning property). The removed element cannot be part of a loop, thus the remaining elements still contain a loop. The contradiction lies in the following:

- The remaining array does not contain an empty row or column (as the original did not either and the column [row] of the removed element contains an assigned element because of the dependency).
- Every tied element can be reached from any other one, as the omitted element can be left out from the middle of any chain, because it was the only element in its row [column], and this means that its preceding column-adjacent [row adjacent] element can directly be followed by its succeeding column-adjacent [row adjacent] element.

The properties are fulfilled, thus, as the critical number of the original array is minimal, the remaining elements would form a basis, though they contain a loop.

Thus the condition given in the statement is necessary and sufficient for q elements to form a basis in the array. (The “loop-free” condition formulated in the definition is difficult to check.) The “spanning” property is very easy to check, and for checking the “dependent” property, notice that it is sufficient to reach all tied elements along chains starting from an arbitrary element. As neither of the two properties can be deduced from the other, the basis could be defined by them. A *basis* of an $m \times n$ array is a spanning and dependent set of $m + n - 1$ tied elements.

Using the two properties of a basis formulated in lemma 1.7 it is easy to prove an important theorem of this chapter, which will be applied when finding the optimal program by necessary changes.

Theorem 1.10: An $m \times n$ array with a basis is given. There exists one and only one loop along tied elements starting from any free (i.e. not tied) element.

Proof: Take an arbitrary free element. Both its row and column contain a tied element according to property 1. (If there are more than one, choose any of the adjacent tied elements for the time being.) Take the chain connecting these two

tied elements and close it with the free element. According to lemma 1.3, this closed chain can be reduced to a loop through the free element.
It still has to be proved that this loop is unique. Suppose indirectly that there are two different loops along tied elements through this free element. Now a loop can be constructed along the tied elements of the basis by the following method, to provide a contradiction. Start from the free element along one of the loops in (e.g.) column-direction. If there are such, go on along the common tied elements of the two loops. Start listing the elements of the loop to be constructed with the first not common element, and continue it until the next common element (until the last element of the loop gone through at the latest). From this, turn back along the other loop and list its elements in reverse order until the point of separation. The connection of the parts of the two loops is adequate at both ends, thus the constructed closed chain is a loop.

The above theorem and its two following consequences are to be applied in the third chapter.

Lemma 1.11: If at least $m+n$ elements are given in an $m\times n$ array, then they must contain a loop.

Proof: Let us take the tied elements in an arbitrary order. At each step it is to be checked whether the set of elements already taken contains a loop. If it does, the proof is done, otherwise let us go on, until the number of elements is $q = m + n - 1$. If the set still does not contain a loop, then the elements form a basis. In this case, according to the previous theorem, these elements, together with any other element, contain a loop.

Lemma 1.12: If there are less then $q = m + n - 1$ loop-free tied elements in an $m\times n$ array, then these elements can be completed to form a basis.

Proof: The proof is done by induction, applied to the critical number. In case of a 2×2 array (i.e. when $q = 3$) it is obvious that any one or two elements can be completed to a three-element basis. Suppose that the statement is true in case of $q = r$ ($r \geq 3$). Let us take an array for which $q = r + 1$, and let us take less then q loop-free tied elements in it. Omit an empty or a one-element row [or column]. (There surely exists a row or column with at most one tied element. It can also be achieved for the remaining array to have at least two rows and columns.) The critical number of the array obtained is $q = r$, thus less then r loop-free tied elements can be completed to a basis in it. If one-element row [or column] was omitted, then the remaining tied elements in the remaining array can be completed to an r-element basis, as their number is less then r. Adding the omitted row [or column] with its omitted element, we get a basis for the original array. If the omitted row [or column] was empty, then the procedure is the same; insert back the omitted row [or column] with one of its elements made tied. (In this case it can happen that the in remaining array, there are r tied elements, thus the completion is unnecessary.)

In connection with bases it is important to note that in an $m \times n$ array there is a finite number of different bases, however their number can be quite large in case of big arrays. It is because the number of selections of $q = m + n - 1$ elements from $m \cdot n$ elements of an array is

$$\binom{m \cdot n}{m+n-1},$$

and not all of the selections are loop-free. Hereunder we will use only the fact that the number of bases is bounded from above. (The precise number is given in the appendix.)

2 The Initial Program

In the followings the definition of a transportation problem is needed.

Definition 2.1: In case of a *transportation problem m* stores and *n* destinations are given, and the goods have to be taken from the stores to the destinations such that the cost of transporting has to be minimal. The stocks of the stores, the demands of the destinations and the cost of the routes for a unit quantity of goods are known.

Suppose the followings:
- Solid goods are measured in loads and the cost has to be paid for each load. The model can also be used for the optimization of transportation of gas or liquid through run of pipes or transportation of energy.
- The totals of stocks and demands are equal, thus all the stocks will be delivered and all the demands will be satisfied. (This condition can easily be released after solving the main task.)

Formally the unit costs of the transportation (from here: transportation costs) are given by an $m \times n$ array, where rows represent stores, columns represent destinations. Stocks are written at the end of the rows, demands are written at the bottom of the columns. This set of data is called an $m \times n$ “transportation table”. In the solution the routes along which the transportation is made are framed, and the numbers of deliverable units are written above them.
In the followings the mathematical model of the problem is given. (Instead of matrices we shall continue to use arrays.)

Definition 2.2: In a *transportation problem* an $m \times n$ matrix is given whose entries are non-negative real numbers; there are also given positive numbers corresponding to each row and column (the “values” of the rows and columns), where the sum of the values of the rows is equal to the sum of the values of the columns. Let a real number be corresponded to each entry of the matrix such that

1) the sums of the numbers in the rows and the columns are the previously given value;
2) the numbers are non-negative;
3) the sum of the products of the entries of the matrix and their corresponding values is minimal.

Out of the conditions 1) is called *restrictive condition*, 2) is called *non-negativity condition*, 3) is called *optimising condition*. Correspondences satisfying the first two conditions are called *possible solutions*; among them *optimal solution* is one that satisfies the third condition. The sum of the products drafted in condition 3 is called *objective function*; the problem involves the minimization of this.
In the mathematical model, the corresponded numbers represent the deliverable units. Where there is no transportation, the corresponded number is 0. In a possible solution, those entries of the matrix, whose corresponded number is positive, are called *tied elements*; the others are called *free elements*. The values of the rows are the stocks, the values of the columns are the demands. The possible solutions are also called programs, as they give how much to transport from which store to which destination.

The following theorem shows the connection between bases and possible solutions. The nonzero elements of the possible solution to be constructed (in some cases together with some additional zeros) make up a basis.

Theorem 2.3: Every transportation problem has a possible solution.

Proof: Let us construct a possible solution according to the followings: make an arbitrary element tied and correspond the value of its row or column to it, whichever is smaller, then omit the row [or column] whose value was corresponded. (If the two mentioned numbers are equal, then correspond this value to the tied element, and omit either its row or column, but only one of them.) As the values of the rows and columns are non-negative, the corresponded value to the tied element is also non-negative. Reduce the value of the remaining column [or row] by the corresponded value, so this result is surely non-negative. Continue the procedure with the remaining array. The value corresponded to the last tied element necessarily satisfies both its row and column, thus they both can be omitted. This arises from the fact that the sums of the rows and columns are equal. Finally let us correspond 0's to the free elements. This method satisfies the restrictive condition and the non-negativity condition, too.
(Tied elements can be chosen arbitrarily,as their number is exactly q and they do not span a loop.)

We remark that it can happen in the above construction that 0 is corresponded to a tied element so it would turn to free. (This can happen if a common element of a row and column whose values are equal, is made tied. In this case a 0-valued row or column will remain.) Because of reasons detailed later these cases are called *degenerated*, and exceptionally 0-valued tied elements are allowed.

To solve a transportation problem by distribution, first an initial program is set up, which is improved step by step in order to find the optimal solution. It can be seen in the above proof that a possible solution built on a basis can be found very easily. The fact, that the algorithm allows much freedom, can be used to construct an initial program with a small-valued objective function so less improvement is necessary. Let the (arbitrarily chosen) tied element be the least possible in the first approach.

Of course it is not evident why we need to make a basis (i.e. exactly *q* elements) tied to give a possible solution to a transportation problem. The explanation of this is given in the following chapter.

3 Steps of Improvement

Lemma 3.1: Suppose that a transportation problem has a possible solution such that the tied elements contain a loop. Now the solution can be reprogrammed such that the value of its objective function does not increase (in general it decreases).

Proof: Only values belonging to the 2*k*-element-loop will be modified according to the following algorithm. Start from an arbitrary element and go round the loop, for instance in the direction of the chosen element's row-adjacent. Let the entries of the array (i.e. the transportation costs) be: $\alpha_1, \beta_1, \alpha_2, \beta_2, \ldots, \alpha_k, \beta_k$; and the corresponding numbers in the given possible solution (i.e. the deliverable units) be: $x_1, y_1, x_2, y_2, \ldots, x_k, y_k$. Now the value of the objective function (i.e. the total cost of delivery) is

$$A + \sum_{i=1}^{k} \alpha_i x_i + \sum_{i=1}^{k} \beta_i y_i \ ,$$

where *A* is the contribution of the other tied elements to the objective function.

Let $\sum_{i=1}^{k} \alpha_i$ denote α and $\sum_{i=1}^{k} \beta_i$ denote β.

Suppose that $\alpha > \beta$. Find the minimum of corresponded numbers $x_1, x_2, \ldots, x_k$, and chose a positive *t* not greater than this value. Decrease the number belonging to the starting element by *t*, then, according to the previous circle of the loop, increase and decrease the elements by turns, until we return. This method ensures that the values of every row and column are still satisfied, as the same number is subtracted from one of the terms, and added to the other term of the sum. Besides, by choosing the value of *t* it is ensured that none of the values turns to negative, they may be 0. Thus the new program is also possible. The value of the objective function is

$$A+\sum_{i=1}^{k}\alpha_i(x_i-t)+\sum_{i=1}^{k}\beta_i(y_i+t) = A+\sum_{i=1}^{k}\alpha_i x_i+\sum_{i=1}^{k}\beta_i y_i-\sum_{i=1}^{k}\alpha_i t+\sum_{i=1}^{k}\beta_i t =$$
$$=\left(A+\sum_{i=1}^{k}\alpha_i x_i+\sum_{i=1}^{k}\beta_i y_i\right)-t(\alpha-\beta).$$

As $\alpha>\beta$, in case of a positive *t*, the value of the initial program has actually decreased.
If $\alpha<\beta$, then the value of *t* is chosen such that it is not greater than the minimum of numbers $y_1, y_2, \ldots, y_k$. In this case the value belonging to the starting element is increased, the next one is decreased, and the procedure is continued so by turns. Thus the value of the objective function is

$$A+\sum_{i=1}^{k}\alpha_i(x_i+t)+\sum_{i=1}^{k}\beta_i(y_i-t)=\left(A+\sum_{i=1}^{k}\alpha_i x_i+\sum_{i=1}^{k}\beta_i y_i\right)-t(\beta-\alpha)$$

which is less then the original one.
If $\alpha=\beta$, or if the value of *t* is exceptionally 0 in case of any relation (because this is the minimal value), then the value of the objective function does not change. Naturally in the latter case the program is not modified.
It is important to note that if *t* is equal to the minimum of the examined numbers, then there will surely be one or more 0's among the new corresponding values. These elements will be considered either free or tied, as expedient.

There are two important applications of the above lemma. First it will be shown that it is sufficient to search for the optimal solution among possible solutions built on a basis. The next statement points at this fact.

Theorem 3.2: Suppose that there is a possible solution of a transportation problem, which is not built on a basis. Now another possible solution can be constructed from this, such that it is built on a basis and the value of its objective function is not greater than the original.

Proof: If there is a loop among the tied elements then one (or more) of its elements can be made free with the above procedure. (The actual value of *t* has to be chosen always as the minimum.) This has to be continued until there is no more loop. These steps do not increase the value of the objective function. According to lemmas 1.11 and 1.12, the number of the remaining loop-free tied elements is at most the critical number of the array and can be completed to form a basis. Let us correspond 0 to the new tied element (if they exist) but consider them as tied. With this the value of the objective function does not change. Thus a possible solution built on a basis is constructed, whose objective function-value is not greater than the original.

We see that it is practical to allow such tied elements whose corresponded values are 0, so it is sufficient to examine the possible solutions built on a basis. It cannot be excluded that the optimal solution itself is degenerated. From now on every possible solution will be built on a basis in this paper, otherwise we will mention. Using the same method, steps of improvement can be executed, which is called *distribution*. In the *basic step of distribution* we should examine all free elements whether they are worth taking in the given program so that a tied element is changed by them; at the end such a possible solution is obtained whose objective function-value is not greater than the original. Three cases can occur. The program is either improved (i.e. a solution is constructed whose objective function-value is less than the original), or it is not improved either by retaining the original solution or constructing a new solution whose objective function-value is equal to the original.

In the first chapter it was proved (theorem 1.10), that there exists one and only one loop along tied elements for any free element. Build this loop to the examined free element then proceed according to the algorithm described in the proof of lemma 3.1. Let the free element be chosen as starting element. If $\alpha > \beta$, then during the re-programming, the value belonging to the starting element was decreased. This is not appropriate for the free element to be taken in, as the corresponded value is 0, this can only be increased. (From now on we shall skip detailing the case when $t = 0$ as it does not result in any changes in the program.) Thus the free element can be taken in the program if and only if $\alpha \leq \beta$. The value of *t* is chosen as the minimum of numbers $y_1, y_2, \ldots, y_k$, so it can be achieved that (at least) one value corresponding to a tied element is 0 in the new possible solution. This (or any one of these) changes to free, while the originally free element changes to tied. The objective function-value of the constructed possible solution is less than, or, in case of $\alpha = \beta$, is equal to, the original.

If none of the free elements is worth taking in, then the given possible solution cannot be improved locally (i.e. by changing a tied element). For the present it is not clear yet whether the examined program is optimal, or it is possible to find another possible solution with other tied elements, whose objective function-value is less. We shall answer this question in the next chapter.

4 The Optimal Program

If a possible solution is given, we should find a unique loop along tied elements for any free element, then after relation-check, the possible decrease of the objective function should be calculated. Finally the free element should be chosen and taken in the program with which the decrease is the greatest. However, in case of arrays of big size it is a long and tedious work to find the mentioned loop. This problem is simplified by the *method of potentials*.

Definition 4.1: Let an *m*×*n* transportation table be given with a possible solution. Correspond a value *u* to any row and a value *v* to any column. Values

$u_1, u_2, \ldots, u_m$ and $v_1, v_2, \ldots, v_n$ are called *potentials* belonging to the possible solution, if it is true for any tied element that it (i.e. the transportation cost) is equal to the sum of its row's and its column's potential.

In the above definition the indices are not necessarily the same as the indices of the matrix. Potentials can belong to rows and columns, but (as it is seen in the following statement) also to tied elements.

Lemma 4.2: Potentials can be constructed for every possible solution.

Proof: It is going to be proved that starting from any tied element, potentials can be assigned. As shown in the first chapter, there is a chain between any two tied elements, thus any tied element can be reached along tied elements from the starting one (from the "root"). These chains starting from the root form a tree, with main-branches and possibly side-branches. Choose the potentials of the root (i.e. of its row and column) arbitrarily, according to the definition. First, assign the missing row- or column potentials to all elements along one of the main-branches such that it satisfies the conditions described in the definition. (One of them already exists, namely the one from where we got to the element.) This will be the potential of the given row or column. An element which has not come up yet, cannot have both of its potentials assigned, because this would mean a closed chain along tied elements. If we have gone along a branch, continue along another one from the connecting point of the previous one. Thus, finally every tied element is reached. As a result, a potential is assigned to every row and column, as the root gets two, every other element gets one, a total of $m+n$.

Now let us see the use of potentials. It has been stated that a free element can be taken in, if and only if $\alpha \le \beta$, and a program with lower cost can be obtained if $\alpha < \beta$ (and $t > 0$). Here α_1 is the free element and the others (now starting in row-adjacent direction) are the transportation costs of the tied elements of the corresponding loop, respectively. Consider an arbitrary free element and a corresponding 2*k*-element loop. Suppose that the row- and column potentials of the tied elements in this loop, starting from the free element in row-adjacent direction, are the following: $(u_1;v_1), (u_2;v_1), (u_2;v_2), \ldots, (u_k;v_k)$. (This can be obtained by rearranging indices. It was taken into consideration that the second loop-element is in the same column as the first, the third is in the same row as the second, and so on.) Now

$$\beta_1 = u_1 + v_1,\ \alpha_2 = u_2 + v_1, \ldots, \beta_k = u_k + v_k .$$

Thus

$$\sum_{i=1}^{k} \alpha_i = \alpha_1 + \alpha_2 + \ldots + \alpha_k = \alpha_1 + (u_2 + v_1) + \ldots + (u_k + v_{k-1}) = \alpha_1 + \sum_{i=2}^{k} u_i + \sum_{i=1}^{k-1} v_i ,$$

$$\sum_{i=1}^{k} \beta_i = \beta_1 + \beta_2 + \ldots + \beta_k = (u_1 + v_1) + \ldots + (u_k + v_k) = \sum_{i=1}^{k} u_i + \sum_{i=1}^{k} v_i .$$

Using the above equalities $\alpha < \beta$ holds if and only if

$$\alpha_1 + \sum_{i=2}^{k} u_i + \sum_{i=1}^{k-1} v_i < \sum_{i=1}^{k} u_i + \sum_{i=1}^{k} v_i \text{ , i.e. } \alpha_1 < u_1 + v_k \text{ .}$$

Notice that the free element with transportation cost α_1 is in the row of index 1 and in the column of index *k*. (Here indices mean ones of the potentials.) Our result thus means that a free element has to be taken in the program if and only if its transportation cost is less than the sum of its row- and column potentials. Checking this for all free elements is much simpler and shorter than searching for loops. Assigning potentials has to be done only once and it is not laborious.

Checking whether a possible solution can be improved has to be done as follows. Give an arbitrary potential-assignment of the program. This can be done by choosing the row-potential of an arbitrary tied element as 0, and the others are distributed according to the algorithm described in the proof. After this, construct an $m \times n$ array, in which each transportation cost is decreased by the sum of its row- and column potentials. (Now of course, there are 0's in place of the tied elements.) By taking the free elements in the program, in place of which there are non-negative numbers, the total transportation cost would not be decreased. Thus, if all elements are non-negative in this array, the program cannot be improved by distribution. We note that if there is such a 0 which stands in place of a free element, then another solution, whose cost is the same as the original's, can be constructed by taking the free element in the program.
If a decrease may be produced by the replacement of several free elements, then replace the element producing the largest decrease, i.e. the one for which the difference $t(\beta - \alpha)$ is the biggest.

What guarantees that by basic steps of distribution, such an array can be obtained which cannot be further improved? Let us settle that in the stage of the algorithm when an optimal array is looked for, the existing program is only changed if the objective function-value of the program is less then the previous. (Finding programs with the same objective function-value has significance only when the program has to be modified, i.e. when alternative optimal programs are looked for.) Thus it can be stated that every array is different from the others (as their objective function-values are different). As an array has a finite number of different bases, it is not possible to construct infinite different possible solutions (built on bases), i.e. the improving procedure ends in finite steps.

It is still to be proved that a program which cannot be improved locally is globally optimal. As much freedom is left when setting up the initial program, and the further progress is not unique, it is not trivial that the optimal solution is obtained when there is no more chance to improve. Is not it possible that there is an entirely different program with lower cost? (We have seen that in case of arrays there are quite a many bases.)

Theorem 4.3: If a possible solution cannot be improved by taking in any of the free elements then it is optimal.

The short summary of the proof is the following. Let a possible solution (built on a basis) be given. Possible solutions are called adjacent if they are obtained from each other by changing a free element (irrespectively of improvement). We shall construct such possible solutions (not built on a basis) that are in some sense "close" to the given program. We mean by "closeness" that the delivered quantities are not very different in the two programs. We shall prove that if the adjacent solutions are all worse (or not better) than the initial program then none of the "close" programs can be better with respect to cost reduction. Thus we shall get to a contradiction, because it will be proved as well, that if there is a better solution than the initial one then there is a better "close" program too. The appropriate "close" programs will be constructed by combining the original program and one or more other solutions.

The following lemma, which is significant itself too, will be used in the proof.

Lemma 4.4: If values programmed on free elements of an arbitrary basis are equal in two possible solutions of a transportation problem then values programmed on tied elements of this basis are equal too (i.e. the two programs are identical).

Proof: As seen in the first chapter (lemma 1.7), there exists such a row [or column] which contains exactly one tied element, thus the value programmed on this element is determined by the row- [column-] sum. By omitting this row [column] the procedure can be continued similarly until all tied elements are reached. Thus if the above condition is satisfied, then all values programmed on tied elements in the basis are uniquely determined.

Let us now prove the theorem for the optimal program.

Proof (of theorem 4.3): Denote the programs by block capitals (*A*, *P* etc.), and their total transportation cost by *z(P)*. Factors *λ* are all non-negative. Let *λP* be the program which is obtained by multiplying every quantity in *P* by *λ*; let *P+R* be the program which is obtained by adding the appropriate delivered quantities. It is obvious that in the first case the row-sums, the column-sums and the total transportation cost will be the *λ*-multiple of the original; in the second case these quantities will be added. Because of the fact concerning row- and column-sums, if $P_1, P_2, \ldots, P_k$ are possible solutions of a transportation problem, then in case of $\lambda_1 + \lambda_2 + \ldots + \lambda_k = 1$, their linear combination $\lambda_1 P_1 + \lambda_2 P_2 + \ldots + \lambda_k P_k$ is a possible solution too, because the row- and column-sums will be as specified. On the other hand

$$z(\lambda_1 P_1 + \lambda_2 P_2 + \ldots + \lambda_k P_k) = \lambda_1 z(P_1) + \lambda_2 z(P_2) + \ldots + \lambda_k z(P_k)\ .$$

Let *P* be a possible solution that cannot be improved by distribution. Let the adjacent programs be (according to an arbitrary order of free elements):

$R_1, R_2, \dots, R_s$. (Now $s = mn - (m+n-1)$ and $z(R_i) \geq z(P),\ 1 \leq i \leq s$.) Suppose that $\lambda_1, \lambda_2, \dots, \lambda_s$ are small positive real numbers (between 0 and 1, close to 0). Set up the program $A = \lambda_1 R_1 + \lambda_2 R_2 + \dots + \lambda_s R_s + (1 - \lambda_1 - \dots - \lambda_s)P$, that is "close" to *P*. Its transportation cost is not less than *P*'s:

$$z(A) = \lambda_1 z(R_1) + \lambda_2 z(R_2) + \dots + \lambda_s z(R_s) + (1 - \lambda_1 - \dots \lambda_s) z(P) \geq$$
$$\geq \lambda_1 z(P) + \lambda_2 z(P) + \dots + \lambda_s z(P) + (1 - \lambda_1 - \dots - \lambda_s) z(P) = z(P).$$

What are the deliverable quantities of this possible solution *A*? We will use the fact that an arbitrary free element in program **P** is free in all of its adjacent programs as well, except in the adjacent program which is obtained by taking this element in. Thus values programmed on the original free elements in *A* are $\lambda_1 h_1, \lambda_2 h_2, \dots, \lambda_s h_s$, where $h_1, h_2, \dots, h_s$ are values programmed on the original free elements in $R_1, R_2, \dots, R_s$.

Now suppose indirectly that there exists a possible solution *Q* which is better than *P*, i.e. its transportation cost is less ($z(Q) < z(P)$). Construct program $B = \lambda Q + (1 - \lambda)P$, which is also "close" to *P*. Let *λ* be a small (between 0 and 1, close to 0) positive real number again. The transportation cost of *B* is less than that of *P* because

$$z(B) = \lambda z(Q) + (1 - \lambda) z(P) < \lambda z(P) + (1 - \lambda) zP = z(P).$$

Values programmed on the original free elements in *B* are $\lambda g_1, \lambda g_2, \dots, \lambda g_s$, where $g_1, g_2, \dots, g_s$ are values programmed on the appropriate elements in *Q*, respectively.

We state that numbers $\lambda_1, \lambda_2, \dots, \lambda_s$ and *λ* can be chosen such that *A* is identical with *B*. Let us make the values programmed on the free elements of *P* equal:

$$\lambda_1 h_1 = \lambda g_1, \dots, \lambda_s h_s = \lambda g_s \text{, i.e let } \lambda_1 = \frac{\lambda g_1}{h_1}, \dots, \lambda_s = \frac{\lambda g_s}{h_s}.$$

(For the time being we suppose that the divisions are sensible, i.e. $h_i \neq 0,\ 1 \leq i \leq s$.) If *λ* is small enough, then the obtained positive terms will be small enough for their sum to be small, or at least not greater than 1. According to lemma 4.4 it can be stated that programs *A* and *B* are identical and this leads to a contradiction because of the inequalities concerning transportation costs ($z(A) \geq z(P) > z(B)$).

The above proof does not work if $h_j = 0$ for any *j* ($1 \leq j \leq s$). This can happen if there is a degeneration in *P*, and when taking in the *j*th free element (call *x*), such a tied element gets out of the program whose corresponding value is 0. There is no problem if this free element of *P* is a free element in *Q* as well, because in this case the equality $\lambda_j h_j = \lambda g_j$ can be made true by any λ_j. But if this element is tied in *Q*, the proof has to be modified. Let μ be an arbitrary positive number. Instead of possible solution R_j consider program R'_j in which the value programmed on *x* is μ, and along the loop belonging to *x* in *P*, values programmed on tied elements are decreased and increased by μ by turns. Thus R'_j satisfies the conditions concerning row- and column sums, but it is not a solution built on a basis. What is more, it is not a possible solution because of the decreases along the loop, as the corresponding value of one (or more) of the tied elements of *P* will be negative. The inequality $z(R'_j) \geq z(P)$ holds because R'_j (just like the other adjacent programs) was obtained from *P* by "worsening" along a loop. (*P* is locally optimal, thus cannot be improved along a loop.) Use R'_j instead of R_j, and the proof will work in this degenerated case: because of $h_j = \mu$ the division is sensible. It will not cause a problem that there are negative corresponding values in R'_j, because the linear combination will remove them. Values programmed on the free elements of *P* are positive in program *A*, and they are equal to the appropriate values of *B*, thus the two programs are identical. As *B* is surely possible, *A* must be possible too. If more *h* values are zero, then the appropriate adjacent programs are exchanged in the proof as described here.

Here we remark and interesting but not trivial observation, that has a practical background as well. Suppose that stocks and demands are integers in the transportation problem. This is quite natural when solid goods are transported in loads or when wagons have to be directed to other stations. In this case the delivered quantities of the optimal program are required, and will actually be received, as integers. This is explained by the fact that during setting up the initial program and during the improvement, a minimum of integers are looked for, or integers are added to, subtracted from, each other; thus integers, as results of these operations, are corresponded to the routes. As a result of these steps, the optimal program is obtained.

5 Supplements

In order to make the presentation of transportation problems complete, two more methods need to be shown, though these do not contain anything new as compared to the usual investigation.
Especially in case of manual calculation, the reduction of transportation costs is useful. This method is based on the following statement.

Lemma 5.1: The basis, offering an optimal solution of a transportation table, is not changed if each element of an arbitrary row or column is decreased by an arbitrary real number; being only careful that no negative value should appear.

Proof: Suppose that the value of the row [or column] is *s*. Let the values of the tied elements in the row [or column] be $\delta_1, \delta_2, \ldots, \delta_k$, and the appropriate corresponding values be $w_1, w_2, \ldots, w_k$ $(w_1 + w_2 + \ldots + w_k = s)$. The objective function-value is $z = A + w_1\delta_1, + w_2\delta_2 + \ldots + w_k\delta_k$, where *A* is the contribution of the other tied elements to the objective function. If *γ* is subtracted from all elements of the row [or column] then the objective function-value of this possible solution in the new transportation table is $z' = A + w_1(\delta_1 - \gamma) + \ldots + w_k(\delta_k - \gamma) =$ $= A + w_1\delta_1 + w_2\delta_2 + \ldots + w_k\delta_k - \gamma \cdot s$, i.e. it differs from the original only in a constant (independent of the possible solution). This means that the same possible solution is the optimal, only the objective function-value is reduced by $\gamma \cdot s$.

Using the above method more times in succession it can be achieved that the transportation costs are much less and calculation is simplified. Go through each row then each column and reduce the transportation costs of the given row [or column] by its minimum. Thus there will be 0 in every row and column which, in case of smaller tables, can help us to find an optimal solution. (If we “notice” a basis, where all tied elements are 0, then this can obviously lead to an optimal solution, because in this case the value of the objective function is 0, and it cannot be less than that. It only has to be checked whether this solution is possible or not.) This method is called *reduction* and it can be done in arbitrary order.

After defining the transportation problem, it was assumed that the total of the stocks is equal to the total of the demands. This fact was used during setting up the initial program, as the corresponding value of the lastly chosen element satisfied both the values of its row and its column. However, this assumption is too rigid when thinking of applications. Luckily, this problem can be solved easily. If the total of the stocks is more than the total of the demands, then the difference stays where it is, and naturally this does not increase transportation costs. Take up a *fictive destination*, i.e. a new column. Let its demand, i.e. the value of the column, be the difference; and let the transportation costs of the corresponding routes, i.e. the elements of the column, be 0. Thus it is achieved that the table satisfies the original conditions, and the optimal solution received by distribution also gives which stocks of goods have to be delivered to the fictive destination. (These will not be moved.) If the total of the demands is more than the total of the stocks, then, similarly, *fictive stores* are taken up, i.e. a new row. Of course, stocks arriving from the fictive stores mean unsatisfied demands.

We note that if a fictive destination is taken up then reductions can only be made in columns, as there are 0’s in every row; or in case of fictive stores, reductions can only be made in rows. It is important, that during the solution, inserting fictive places has to be the first step, as later the equality of the sums of row- and column-values is necessary.

6 Appendix: Number of Bases

The number of bases of an $m \times n$ array will be given in this chapter. The statements of the first chapter will be used and another property of bases will be needed as well.

<u>Lemma 6.1</u>: If $m+n-1$ elements of an $m \times n$ array are given with the next property, then these elements form a basis: rows and columns can be omitted one at a time, such that always a one-element row or column is removed (together with its element).

<u>Proof</u>: It will be proved that the set of the elements cannot contain a loop. In the row and column of any element of a possible loop there is at least one more element, namely its adjacent in the loop, thus this element cannot be removed (neither its row nor its column), until the loop exists. The loop-adjacent of a loop-element cannot be removed either, because this could only be done when its other loop-adjacent has been removed. And so on, going along the elements of the loop, it can be seen that a loop element can be removed only if it has already been removed, which is impossible.

<u>Lemma 6.2</u>: In any basis of an $m \times n$ array
- in case of $m \geq n$ there are at least $m-n+1$ one-element rows;
- in case of $n \geq m$ there are at least $n-m+1$ one-element columns.

Before proving the lemma we note that in the special case of $m=n$ it states that the bases of quadratic tables contain one element rows and columns as well.

<u>Proof (of lemma 6.2)</u>: To prove the first statement, suppose that there are *k* one-element rows and the other $m-k$ rows contain at least two elements. Thus the number of elements is $q=m+n-1 \geq k+2(m-k)$. From this we get $k \geq m-n+1$. It has to be emphasised that as $m-n+1 \geq 1$, in case of $m \geq n$, there must be at least one one-element row. The other statement can be proved by exchanging rows and columns (*m* and *n*).

<u>Theorem 6.3</u>: In an $m \times n$ array, there are $n^{m-1} \cdot m^{n-1}$ different bases.

<u>Proof</u>: A one-to-one correspondence will be given between the bases of an $m \times n$ array and the $(m-1)+(n-1)$-element sequences whose first $m-1$ elements are from the set $\{1, 2, \ldots, n\}$ and the further $n-1$ elements are from the set $\{1, 2, \ldots, m\}$. The number of such sequences is obviously $n^{m-1} \cdot m^{n-1}$. The first of the following methods uniquely corresponds such a sequence to every basis, the second uniquely corresponds a basis to every sequence. This means that the number of bases is also $n^{m-1} \cdot m^{n-1}$.
The proof is only done for those arrays which have at least as many rows as columns. This can be done because the number of bases does not change if rows and columns of an array are exchanged (i.e. the equivalent matrix is transposed).

According to the previous statement, in case of $m \geq n$ there are at least $m-n+1$ one-element rows. The first step of the procedure is to remove the first $m-n$ one-element rows according to their increasing row-indices and record the column-indices of the omitted elements. Now a quadratic, $n \times n$ array is obtained. (In case the original array is quadratic, the previous step has to be left out.) This must contain one-element rows, out of these remove the one with the least row-index, record its column-index after the previous recorded indices. The obtained array has one more columns than rows, thus it must contain one-element columns, out of these remove the one with the least column-index and record its row-index (apart from the column-indices). Now, again, a quadratic array is obtained, so the procedure is continued until the array is "consumed". The last remaining element does not have to be recorded. We emphasise the followings:

- If more than one rows [or columns] can be removed, always remove the one with the least row- [or column-] index.
- Always record the other index, one after the other, keeping apart the two types.
- The removed elements do not have to be taken into consideration, when one-element rows and columns are looked for, but the indices are recorded according to the original array, not to the reduced one.

Finally write the row-indices of the removed columns after the column-indices of the removed rows. As the last element is left in its place, $m-1$ rows and $n-1$ columns are removed altogether. The column-indices are from the set $\{1, 2, \ldots, n\}$, the row-indices are from the set $\{1, 2, \ldots, m\}$, thus the sequence we get is indeed of the required form. The steps are unique according to the rules in case of any basis, thus the correspondence is unique.

Now it will be shown how it is possible to "find" the basis for an arbitrary sequence of this type, to which it was corresponded, i.e. how it is possible to find the pair of the recorded indices, and finally the indices of the remaining element. First the order of removal of elements is prepared. Write the first $m-n$ column-indices at the beginning, then (continued with a column-index) the column- and row-indices, by turns. Leave space for the appropriate index-pairs, and for the remaining element with unknown indices at the end. As, together with the row of the remaining element, all rows were removed, the missing row-index pairs of the column indices and the row-index of the remaining element are a permutation of elements $1, 2, \ldots, m$. According to the previous procedure it is obvious that if a row has been removed then its row-index cannot appear later (among the row-indices of the removed columns). Distribute the missing row-indices starting from the beginning, such that always write the least out of the ones that have not been distributed yet and not appearing later, and write the last two remaining indices in increasing order at the end. Similarly, distribute the missing column-indices of the row-indices that are the permutation of elements $1, 2, \ldots, n$. It has to be proved that the obtained elements form a basis and the sequence corresponded to this basis is the one we started with. The procedure shown at the beginning of the proof can be performed on the obtained elements, thus they would be removed exactly in the given order (together with their rows or columns). The reason for this is that when a row is to be removed, then we are exactly at such an element which is alone in its row, and among these, whose row-index is the smallest, as the column-indices were distributed so. The same is true when the next step is the removal of a column. As these removals satisfy the conditions of the first statement, thus these elements form a basis.

The above result can be formulated in two, apparently different forms, which take us to the areas of linear algebra and graph theory.

Theorem 6.4: Consider the set of $(m+n)$-element vectors over the field of real numbers. The number of maximal linearly independent vector sets, that can be chosen out of vectors

$$\mathbf{a}_{\mathbf{i,j}} = \begin{bmatrix} \mathbf{e}_{\mathbf{i}} \\ \mathbf{e}_{\mathbf{j}} \end{bmatrix}, \ (1 \le i \le m, 1 \le j \le n)$$

is $n^{m-1} \cdot m^{n-1}$.

Proof: Let us make a correspondence between the (*i;j*)-indexed element of an $m \times n$ array and vector $\mathbf{a}_{\mathbf{i,j}}$. It will be shown that in case of this correspondence a set of vectors is linearly independent if and only if the corresponding set of elements does not contain a loop. The formulated statement follows from this.
The proof exactly will show that a set of vectors is linearly dependent if and only if the corresponding set of elements contains a loop. Vectors corresponded to the elements of a loop are dependent as their sum taken with alternating signs is **0**. Thus, vectors corresponded to a set of elements containing a loop are dependent too, as this set is an extension of the previous case. To prove the other direction, let a dependent set of vectors be given, and also a linear combination of them resulting in **0**. Take an element $\mathbf{a}_{\mathbf{i_1,j_1}}$ of the set of vectors whose coefficient is positive. Now there must be an element $\mathbf{a}_{\mathbf{i_1,j_2}}$ with a negative coefficient in order that the i_1 th component of the linear combination be 0. But in this case there must be an element $\mathbf{a}_{\mathbf{i_2,j_2}}$ with a positive coefficient in order that the $m+j_2$ th component of the linear combination be 0. If we take the elements one after the other this way, there will come an element that has appeared before as the set is finite. Consider the elements of this closed “circle”. (Elements taken before the closure of the circle can be omitted.) The elements of the array, corresponding to the vectors, form a closed chain whose elements are not in one row or column. Thus the chosen set of elements is not loop-free.

Theorem 6.5: Let a bipartite graph be given whose partitions are vertices $\{1;2;\ldots,m\}$ and $\{1';2';\ldots,n'\}$. The number of its different spanning trees is $n^{m-1} \cdot m^{n-1}$.

Proof: Let us make a correspondence such that the *m* rows of the $m \times n$ array are corresponded to the upper partition of the bipartite graph, the *n* columns of the $m \times n$ array are corresponded to the lower partition of the bipartite graph, and the elements of the array are corresponded to the edges connecting the vertices representing the appropriate rows and columns. This correspondence is one-to-one. Bases of the array are exactly corresponded to the spanning trees of the graph, as these graphs are acyclic and every vertex is an endpoint of one or more edges. A cycle would represent a loop, a vertex with a zero degree would mean an empty row or column.

Starting the proof of the theorem giving the number of bases in a different way, we can find interesting relationships using the final result. This is shown in the following statement. (This time algebra and combinatorics are touched.)

Theorem 6.6: Let *m*, *n* be positive integers, $m > n + 2$. Now

$$m^n = (m-2)^n \cdot \binom{n+2}{2} - (m-3)^n \cdot 2 \cdot \binom{n+2}{3} + \ldots \pm$$

$$\pm (m-(n+2))^n \cdot (n+1) \cdot \binom{n+2}{n+2}.$$

This statement gives a relationship between powers of the same exponent whose bases are successive integers. E.g. in case of $m = 10, n = 3$ we get:

$$10^3 = 10 \cdot 8^3 - 20 \cdot 7^3 + 15 \cdot 6^3 - 4 \cdot 5^3.$$

Proof (of theorem 6.6): The number of bases is to be determined by mathematical induction. Consider only arrays which have more rows than columns. i.e. $m > n$. In this case there is a one-element row in any basis, and it is also true that there is such a row in bases of arrays with one less rows. Construct all the bases of an $(m+1) \times n$ array by inserting a one-element row somewhere before the first one-element row of the $m \times n$ array. Thus all the bases of the $(m+1) \times n$ array are obtained exactly once from that basis of the $m \times n$ array which can be obtained by omitting the first one-element row of the previous one; and insert the appropriate one-element row to the appropriate place. The new row has to be inserted before the first one-element row because there can be more than one one-element rows in a basis, and thus we can avoid counting some "constructions" more than once. In this case, however, it is necessary to determine the number of such bases of an $m \times n$ array in which the first one-element row stands at a given place.

Denote by $P(m,n)$ the number of bases of an $m \times n$ array, and $P(1,n) = 1$. Those bases of an $m \times n$ array which have a one-element first row can be obtained by inserting a one-element row as first in the bases of an $(m-1) \times n$ array. Thus their number is $n \cdot P(m-1,n)$, as the new element can stand at *n* places. Similarly, those bases of an $m \times n$ array which have a one-element second row can be obtained by inserting a one-element row as second in the bases of an $(m-1)\times n$ array. But in this case, those bases are counted as well, whose first and second rows are one-element, thus the number of these has to be subtracted, as we want to count only the ones, whose first one-element row is the second row. The subtrahend is determined by inserting two one-element rows as first and second in the bases of an $(m-2)\times n$ array, and by counting these. Thus the number of bases whose first one-element row is the second one is: $n\cdot P(n-1,m)-n^2\cdot P(m-2,n)$. Similarly, and using the logical sieve, the number of bases of an $m\times n$ array, which have a one-element row first at the kth place, is:

$$\binom{k-1}{0}\cdot n\cdot P(m-1,n)-\binom{k-1}{1}\cdot n^2\cdot P(m-2,n)+\binom{k-1}{2}\cdot n^3\cdot P(m-2,n)-$$
$$-\ldots\pm\binom{k-1}{k-1}\cdot n^k\cdot P(m-k,n).$$

The last place where the first one-element row can stand is the nth row because there are at least $m-n+1$ of them. The number of these bases is:

$$\binom{n-1}{0}\cdot n\cdot P(m-1,n)-\binom{n-1}{1}\cdot n^2\cdot P(m-2,n)+\ldots\pm\binom{n-1}{n-1}\cdot n^n\cdot P(m-n,n).$$

A one-element row can be inserted before the kth row in $k\cdot n$ ways, by choosing the place of the new row and of the new element. Finally, the number of bases of the $(m+1)\times n$ array is:

$$P(m+1,n)=n\cdot\left(n\cdot P(m-1,n)\right)+2n\cdot\left(n\cdot P(m-1,n)-n^2\cdot P(m-2,n)\right)+\ldots\pm$$
$$\pm n^2\left(\binom{n-1}{0}\cdot n\cdot P(m-1,n)-\binom{n-1}{1}\cdot n^2\cdot P(m-2,n)+\ldots\pm\binom{n-1}{n-1}\cdot n^n\cdot P(m-n,n)\right).$$

Suppose, by suspicion, that $P(t,n)=t^{n-1}\cdot n^{t-1}$ for any $t\leq m$. For $t=1$ the equality $P(1,n)=1$ is true. Substituting the induction hypothesis we get

$$P(m+1,n)=n^m\cdot(m-1)^{n-1}+2\cdot n^m\cdot\left((m-1)^{n-1}-(m-2)^{n-1}\right)+\ldots+$$
$$+n\cdot n^m\left((m-1)^{n-1}-(n-1)(m-2)^{n-1}+\binom{n-1}{2}(m-3)^{n-1}-\ldots\pm(m-n)^{n-1}\right)$$

Taking the common multiplier n^m out of the parentheses, arrange the sum according to identical powers. Power $(m-k)^{n-1}$ appears first in the kth term, its multiplier is the following with a negative or positive sign (depending on the parity of k):

$$k\cdot 1+(k+1)\cdot\binom{k}{k-1}+(k+2)\cdot\binom{k+1}{k-1}+\ldots+n\cdot\binom{n-1}{k-1}=$$
$$=k\cdot\binom{k}{k}+k\cdot\binom{k+1}{k}+\ldots+k\cdot\binom{n}{k}.$$

As

$$\binom{k}{k}+\binom{k+1}{k}+\ldots+\binom{n}{k}=\binom{n+1}{k+1},$$

the multiplier of power $(m-k)^{n-1}$ is

$$\pm k\cdot\binom{n+1}{k+1}.$$

Finally we get

$$P(m+1,n)=n^m\cdot\left((m-1)^{n-1}\cdot\binom{n+1}{2}-(m-2)^{n-1}\cdot 2\cdot\binom{n+1}{3}+\ldots\pm\right.$$
$$\left.\pm(m-n)^{n-1}\cdot n\cdot\binom{n+1}{n+1}\right).$$

We would like to prove that $P(m+1,n)=(m+1)^{n-1}\cdot n^m$. Compared to the previous, it should be proved that

$$(m+1)^{n-1}=(m-1)^{n-1}\cdot\binom{n+1}{2}-(m-2)^{n-1}\cdot 2\cdot\binom{n+1}{3}+\ldots\pm$$
$$\pm(m-n)^{n-1}\cdot n\cdot\binom{n+1}{n+1}.$$

As statement $P(m+1,n)=(m+1)^{n-1}\cdot n^m$ was already shown in the proof of theorem 6.3, the above identity is necessarily true. Replace *m* by $m-1$ and *n* by $n+1$, thus we get the formula specified in the statement. (During the induction proof it was supposed that $m>n$, and *n* is at least 2; conditions are obtained by reformulating these.)

The previous result leads to some nice identities of combinatorics.

Theorem 6.7: Let $n\geq 3$ be a positive integer. Now

(1) $\binom{n}{2}-2\cdot\binom{n}{3}+3\cdot\binom{n}{4}-\ldots\pm(n-1)\cdot\binom{n}{n}=1$;

(2) $\binom{n}{2}\cdot 2-2\cdot\binom{n}{3}\cdot 3+3\cdot\binom{n}{4}\cdot 4-\ldots\pm(n-1)\cdot\binom{n}{n}\cdot n=0$;

(3) $$\binom{n}{2}\cdot 2^2 - 2\cdot\binom{n}{3}\cdot 3^2 + 3\cdot\binom{n}{4}\cdot 4^2 - \ldots \pm (n-1)\cdot\binom{n}{n}\cdot n^2 = 0\ ;$$

…

(n+1) $$\binom{n}{2}\cdot 2^{n-2} - 2\cdot\binom{n}{3}\cdot 3^{n-2} + \ldots \pm (n-1)\cdot\binom{n}{n}\cdot n^{n-2} = 0\,.$$

Statements (2)-(n+1) can be summarized as follows.
Vector

$$\left(\binom{n}{2}; -2\cdot\binom{n}{3}; 3\cdot\binom{n}{4}; \ldots; \pm(n-1)\cdot\binom{n}{n}\right)$$

is perpendicular to any of the vectors $\left(2^t; 3^t; \ldots; n^t\right)$, where $1 \le t \le n-2$.

Proof (of theorem 6.7): Replace $n+2$ by n in theorem 6.6, thus $n-2$ gets to the place of n. (As n was greater than or equal to 1 ($n \ge 1$), now $n \ge 3$.) Decompose the powers of the right hand side (using the binomial theorem), and compare the coefficients of powers m^{n-2}, m^{n-3}, … m^0 . Now, dividing by a binomial expression, we get the identities formulated above, respectively.

Conclusions

Hopefully the paper was useful for students and teachers dealing with operation research. Nevertheless the aim was also to give an example for a possible bridge over higher and elementary mathematics. (Other similar possibilities are shown at [3].)

References

[1] Wayne L. Winston: Operations Research. Applications and Algorithms
Hungarian edition: Aula Kiadó Kft., 2003

[2] Dr. Csernyák László (szerk): Matematika üzemgazdászoknak,
Operációkutatás II. Tankönyvkiadó, Budapest, 1990

[3] Hraskó András (szerk.): Új matematikai mozaik
TypoTex Kiadó, Budapest, 2002