

---

## **AIMMS Modeling Guide - File Merge Problem**

This file contains only one chapter of the book. For a free download of the complete book in pdf format, please visit [www.aimms.com](http://www.aimms.com) or order your hard-copy at [www.lulu.com/aimms](http://www.lulu.com/aimms).

Copyright © 1993–2011 by Paragon Decision Technology B.V. All rights reserved.

Paragon Decision Technology B.V.	Paragon Decision Technology Inc.	Paragon Decision Technology Pte.
Schipholweg 1	500 108th Avenue NE	Ltd.
2034 LS Haarlem	Ste. # 1085	80 Raffles Place
The Netherlands	Bellevue, WA 98004	UOB Plaza 1, Level 36-01
Tel.: +31 23 5511512	USA	Singapore 048624
Fax: +31 23 5511517	Tel.: +1 425 458 4024	Tel.: +65 9640 4182
	Fax: +1 425 458 4025	

Email: [info@aimms.com](mailto:info@aimms.com)  
WWW: [www.aimms.com](http://www.aimms.com)

AIMMS is a registered trademark of Paragon Decision Technology B.V. IBM ILOG CPLEX and sc CPLEX is a registered trademark of IBM Corporation. GUROBI is a registered trademark of Gurobi Optimization, Inc. KNITRO is a registered trademark of Ziena Optimization, Inc. XPRESS-MP is a registered trademark of FICO Fair Isaac Corporation. MOSEK is a registered trademark of Mosek ApS. WINDOWS and EXCEL are registered trademarks of Microsoft Corporation.  $\text{T}_{\text{E}}\text{X}$ ,  $\text{L}_{\text{A}}\text{T}_{\text{E}}\text{X}$ , and  $\mathcal{A}\mathcal{M}\mathcal{S}\text{-}\text{L}_{\text{A}}\text{T}_{\text{E}}\text{X}$  are trademarks of the American Mathematical Society. LUCIDA is a registered trademark of Bigelow & Holmes Inc. ACROBAT is a registered trademark of Adobe Systems Inc. Other brands and their products are trademarks of their respective holders.

Information in this document is subject to change without notice and does not represent a commitment on the part of Paragon Decision Technology B.V. The software described in this document is furnished under a license agreement and may only be used and copied in accordance with the terms of the agreement. The documentation may not, in whole or in part, be copied, photocopied, reproduced, translated, or reduced to any electronic medium or machine-readable form without prior consent, in writing, from Paragon Decision Technology B.V.

**Paragon Decision Technology B.V. makes no representation or warranty with respect to the adequacy of this documentation or the programs which it describes for any particular purpose or with respect to its adequacy to produce any particular result. In no event shall Paragon Decision Technology B.V., its employees, its contractors or the authors of this documentation be liable for special, direct, indirect or consequential damages, losses, costs, charges, claims, demands, or claims for lost profits, fees or expenses of any nature or kind.**

**In addition to the foregoing, users should recognize that all complex software systems and their documentation contain errors and omissions. The authors, Paragon Decision Technology B.V. and its employees, and its contractors shall not be responsible under any circumstances for providing information or corrections to errors and omissions discovered at any time in this book or the software it describes, whether or not they are aware of the errors or omissions. The authors, Paragon Decision Technology B.V. and its employees, and its contractors do not recommend the use of the software described in this book for applications in which errors or omissions could threaten life, injury or significant loss.**

This documentation was typeset by Paragon Decision Technology B.V. using  $\text{L}_{\text{A}}\text{T}_{\text{E}}\text{X}$  and the LUCIDA font family.

## Chapter 19

### A File Merge Problem

This chapter considers the merging of two statistical database files. The problem can be formulated as a transportation model and solved using a linear programming solver or a specialized network solver. However for large files, the number of variables in the underlying model is too large. To overcome this, a column evaluation approach is proposed. Specifically, a customized solution algorithm which controls the size of the network to be solved by systematically considering a subset of all columns each major iteration. The underlying theory is explained, and subsequently applied to the file merge model.

*This chapter*

The problem and its formulation have been adapted from Glover et al. ([G192]). The underlying theory of the simplex method and column generation can be found in [Ch83].

*References*

Linear Program, Network Program, Simplex Method, Column Generation, Mathematical Derivation, Customized Algorithm, Worked Example.

*Keywords*

---

#### 19.1 Problem description

In this section the problem of merging an income data file and a population data file is described. The structure of these files is examined, and the file merge problem is viewed as a distance minimization problem.

*This section*

Statistical databases are typically developed and maintained in such government institutions as statistical offices, ministries and planning agencies. These databases are the result of extensive surveys involving (tens of) thousands of households or businesses, and contain information that can be used to analyze, for instance, the effect of government policy measures. Examples are the study of welfare measures, the effect of social security benefits or taxation on government income, etc.

*Statistical  
database files  
...*

A statistical database file consists of similar records. Each record has a fixed number of data fields. These data fields contain values that are applicable to a group of households or businesses with similar characteristics. As a result, data is not stored per individual household or business, but aggregated (in some way) for each group. That is why the number of similar households or businesses is always part of each record. The net effect is that the number of records in a file is much smaller than when records are maintained for each individual household or business. Nevertheless, the number of records may still be in the order of thousands. As you will see in later paragraphs, the data field containing the number of families or businesses in each record will play a central role in both the problem and model formulation.

*... and their structure*

One example of a statistical database file used in this chapter is a file referred to as the 'Income Data File' (see [G192]). Each record describes some specific income characteristics of families, and also contains the number of families sharing these characteristics. These families from one record are of course not identical in all respects. For instance, in Table 19.1, the 'Gross Family Income' is an average and thus not exact for an individual family, while the 'Source of Income' is identical for all families in the record.

*Income data file*

Record	No. of Families	Gross Family Income	No. of Family Members	Source of Income	Interest Income
1	20	10,000	3	Commerce	0
2	30	15,500	2	Commerce	1,000
3	25	20,000	5	Agriculture	1,000
4	18	25,000	4	Agriculture	3,000
5	32	15,000	3	Commerce	500
⋮	⋮	⋮	⋮	⋮	⋮

Table 19.1: Income Data File

Another example of a statistical database file used in this chapter is a file referred to as the 'Population Data File' (see [G192]). Each record describes some specific demographic characteristics of families. Again, the families within a record are not identical in all respects. For instance, in Table 19.2, the 'Number of Family Members' is just an indication for the group as a whole, while the 'Head of Household' characteristics may apply to all families in the record.

*Population data file*

Consider the evaluation of a tax reduction scheme. Such a scheme is usually based on a partitioning of households in terms of both income and demographic characteristics. Unfortunately, it is not clear how families in the 'Income Data File' are related to families in the 'Population Data File'. What is needed is a way to combine these two files, so that each new record describes

*Need to merge*

Record	No. of Families	Gross Family Income	No. of Family Members	Head of Household			No. of Family Members Under 18
				Age	Education	Sex	
1	25	15,000	4	40	12	M	2
2	30	15,000	2	25	16	M	0
3	18	20,000	1	30	18	F	0
4	27	25,000	2	35	16	F	1
5	25	20,000	4	25	12	M	1
⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮

Table 19.2: Population Data File

both income and demographic characteristics for an entire group of similar families.

Records in both files contain common information such as ‘Number of Families’, ‘Gross Family Income’ and ‘Number of Family Members’. These data fields form the basis for merging the two files. In practical applications, however, there is not a one-to-one mapping between records in each file on the basis of common characteristics. The database files, presumably derived from the same population, are typically based on data from different samples of households. Merging records in such a case is no trivial exercise.

*Common information*

With database files from different sources, the merging of records becomes somewhat arbitrary. One could even wonder whether entire records should be merged. Perhaps the merging of two particular records should take place for just a subset of families in each record. In any case, some measure has to be developed to determine whether two records are similar enough to be merged. Such a measure can only be based on common information. When the common information is exact, only records with matching entries can be combined. In such a situation, quite a few combinations of records can be ignored. When the common information in a data field is an average or a typical value, then records with differing, but sufficiently close, entries can be combined. One could speak of “distance” between records, where pairs of records with low distance are regarded as similar, and the ones with large distance are regarded as dissimilar.

*Distance between records*

Assume that a distance measure has been determined. Then the goal is to combine records in such a way that the sum of distances for all combined records is minimal. Of course, when two records are combined in this process, it must be decided how many families from each of them share this new record. In addition, the new value of each income and demographic data field must be decided. A specific approach to deal with these questions is proposed in the next section.

*Decisions to be made*

---

## 19.2 Mathematical formulation

In this section the translation of the file merge problem into a network model is proposed and evaluated. The construction of an overall distance function is illustrated for the merging of the ‘Income’ and ‘Population’ data files. In addition, suggestions are made for the computation of data entries that make up the newly constructed records.

*This section*

As noted in the previous section the merging of records is arbitrary, and several approaches can be thought of. Without the use of an optimization model you might think of sorting each of the two files according to one or more major characteristics, and then select families from both sorted files in increasing order to make up new records. Other similar heuristic approaches can be thought of, but they all suffer from the fact that issues are not looked at simultaneously but only sequentially in an ad hoc manner. This is why a mathematical formulation based on simultaneous constraints is proposed in this section.

*Several approaches*

In several modeling exercises the translation of a problem into a model is not obvious at first, but almost trivial or self-evident afterwards. This is also the case with the file merge problem. Each record contains a data field for the number of families, and somehow families in records of the first file must be assigned to families in records of the second file. This sounds like an assignment problem of some sort, which may remind you of the network applications in Chapter 5. As it turns out, the key observation is to view each record as a node in a network.

*Network approach*

Consider the records of one file as supply nodes in a transportation network with the ‘Number of Families’ as the available supply. Similarly, consider the records of the second file as demand nodes with the ‘Number of Families’ as the required demand. Throughout this chapter it is assumed that the total supply of families is equal to the total demand of families. Next consider a decision variable for each pair of nodes (records) to express how many families from a particular supply node will be “shipped” (i.e. assigned) to a particular demand node. You now have the ingredients for describing a set of constraints which contains all permitted assignments of families.

*View as transportation model*

### Indices:

$i$             *supply nodes (records  $i$ )*  
 $j$             *demand nodes (records  $j$ )*

*Transportation constraints*

### Parameters:

$N_i$            *number of families in record  $i$*   
 $N_j$            *number of families in record  $j$*

**Variable:**

$x_{ij}$       number of families shipped from  $i$  to  $j$

**Constraints:**

$$\begin{aligned} \sum_j x_{ij} &= N_i && \forall i \\ \sum_i x_{ij} &= N_j && \forall j \\ x_{ij} &\geq 0 && \forall (i, j) \end{aligned}$$

When the simplex method is applied to the above set of equalities, then the solution values  $x_{ij}$  are guaranteed to be integer as long as both  $N_i$  and  $N_j$  are integer. This (unimodularity) property has been referred to in Chapter 2.

*Integer solutions*

In the context of the file merge problem you may interpret any feasible solution of the above set of constraints as follows. Whenever the variable  $x_{ij}$  is positive, records  $i$  and  $j$  will be merged to form a new record, and the value of  $x_{ij}$  is the number of families sharing that new record. As the total number of families (summed over all records) in both files are identical, all families will be placed in some new record. Note that nothing has been said thus far concerning the contents of the income and demographic data fields of the new records. This will be discussed later.

*Interpretation*

When you add a total distance function as the objective function to the above set of constraints, you obtain a complete optimization model. Any optimal solution of this model states how existing records must be merged to form new records such that the total distance between merged records is minimal. Let  $d_{ij}$  be a parameter containing the distance between records  $i$  and  $j$ . Then the objective function to be added to the above constraints becomes

*Objective function***Minimize:**

$$\sum_{ij} d_{ij} x_{ij}$$

Similarity between records can be stated in a variety of ways. The following formulation of distance was developed and motivated in [G192].

*Formulation of distance***Parameters:**

$G_i, G_j$       'Gross Family Income' in record  $i$  or  $j$   
 $M_i, M_j$       'Number of Family Members' in record  $i$  or  $j$   
 $s_G^2$             estimated variance of all  $G_i$  and  $G_j$  values  
 $s_M^2$             estimated variance of all  $M_i$  and  $M_j$  values

The proposed measure of distance  $d_{ij}$  is then as follows.

$$d_{ij} = \sqrt{\frac{(G_i - G_j)^2}{s_G^2} + \frac{(M_i - M_j)^2}{s_M^2}}$$

Note that by dividing the squared deviation of two data values by their variance, the computed values become comparable in size. Such normalization avoids the situation that deviations in one parameter strongly dominate deviations in another parameter.

Once the solution of the above optimization model is obtained, the number of families for each new record is known. However, the value of the other data fields must still be computed. As stated previously, there is no unique method to determine new data entries whenever the originating records show differing values. Choices have to be made by those who are involved in the construction process. The following constructs are merely two different suggestions. Let the index  $n(i, j)$  refer to a new record derived from records  $i$  and  $j$ . Then the new values of ‘Gross Family Income’ and ‘Number of Family Members’ can be computed as follows.

$$\begin{aligned} G_{n(i,j)} &= (G_i + G_j)/2 && \text{(average)} \\ M_{n(i,j)} &= \max\{M_i, M_j\} && \text{(largest)} \end{aligned}$$

*Constructing  
data fields*

In the file merge model the role of the two database files can be reversed, because the total number of families in each file are assumed to be identical. The question then arises whether such reversal has any effect on the optimal solution. The answer is negative. First of all, any feasible shipments from supply nodes to demand nodes are also feasible shipments from demand nodes to supply nodes. This implies that the set of feasible solutions is not affected by the role reversal of the two database files. Secondly, in the total distance function the coefficients are symmetric for each pair of records. As a result, the optimal solution value remains optimal when the shipping direction between the supply and demand nodes is reversed.

*Choice of origin  
and destination*

An initial guess concerning the size of the newly formed merged file, might lead to the following bounds. Let  $|I|$  and  $|J|$  denote the number of records in file 1 and 2 respectively. Then  $\max\{|I|, |J|\}$  seems to be the smallest number of records in the newly merged file, while  $|I| \times |J|$  seems to be the largest such number. The lower bound is correct, but the upper bound is excessively over estimated. According to the theory of the simplex method (explained in Section 19.4), the maximum number of decision variables away from their bounds is equal to the number of constraints. For the above transportation model this implies that the maximum number of positive decision variables is at most  $|I| + |J|$ . This value is then a much improved upper bound on the number of records in the merged file.

*Size of merged  
file*

Consider the first five records of both the Income Data File in Table 19.1 and the Population Data File in Table 19.2. The total number of families in each of these two portions is 125. Let the distance between records be determined by the differences between 'Gross Family Income'. Then the graph in Figure 19.1 displays a possible merging scheme for which total distance has been kept to a minimum. The number associated with a node is the number of families in the corresponding original record. The number associated with an arc is the number of families in the corresponding new record.

*Example of network ...*

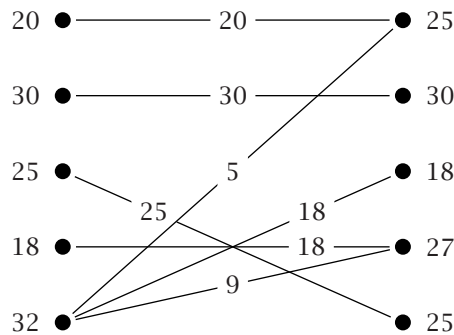


Figure 19.1: Network representation of a solution

On the basis of Figure 19.1 it is now straightforward to construct the merged file as displayed in Table 19.3. As expected, the total Number of Families has remain unchanged. Only the entries underneath the common headers 'Gross Family Income' and 'Number of Family Members' have to be reconciled. In this example, 'Gross Family Income' in the merged file is the average of the entries in the originating files. The 'Number of Family Members' in the merged file is determined differently, and has been set to the maximum of the originating entries.

*... and resulting merged file*

Record	No. of Families	Gross	No. of	Source of Income	Interest Income	Head of Household			No. of
		Family Income	Family Members			Age	Education	Sex	Family Members Under 18
1	20	12,500	4	Commerce	0	40	12	M	2
2	30	15,250	2	Commerce	1,000	25	16	M	0
3	25	20,000	5	Agriculture	1,000	25	12	M	1
4	18	25,000	4	Agriculture	3,000	35	16	F	1
5	5	15,000	4	Commerce	500	40	12	M	2
6	18	17,500	3	Commerce	500	30	18	F	0
7	9	20,000	3	Commerce	500	35	16	F	1

Table 19.3: Merged Data File

---

### 19.3 Solving large instances

This section presents an overview of a method to solve large instances of the file merge model. The key idea is to use an iterative approach that systematically adds and removes variables so that an overall optimized model is found.

*This section*

The number of decision variables and objective coefficients in the file merge model is the product of the two file sizes and consequently the model size can become unmanageable very quickly. When the number of records in each file is of the order  $O(10^2)$ , then the number of decision variables (reflecting all possible pairs of records) is of the order  $O(10^4)$ . Such a model is not considered to be large by today's standards. However, when the number of records in each file is of the order  $O(10^4)$ , then the number of variables is of the order  $O(10^8)$ . In general, models of this size cannot be solved in its entirety using currently available technology. For standard hardware and software configurations, either a special solution approach is required, or the model must be reduced in size prior to being solved.

*Model size*

One approach is to consider whether all possible combinations of records must be included. The number of variables can be reduced significantly if you only consider those variables with a distance value  $d_{ij}$  less than or equal to some sufficiently low cutoff value. Alternatively, a controlled number of variables can be generated if only the  $k$  smallest  $d_{ij}$  values are considered for each  $i$ . There are several other such schemes, all aimed at reducing the number of variables to a manageable level.

*A priori reduction ...*

Using these plausible suggestions many variables can be eliminated, but the question remains whether such a priori reduction will result in unwanted side effects. The answer is surprisingly yes. The main reason is that these reduction schemes are based on the values of  $d_{ij}$  alone, and do not take into account the values of both  $N_i$  and  $N_j$ . In many applications, the reduction schemes discussed above lead directly to infeasible solutions.

*... may not always work*

A better approach to reduce the number of variables would be to carry out the following three steps.

*A better approach*

1. Apply a heuristic to determine at least one feasible solution.
2. Consider some form of a priori reduction.
3. Extend the set of variables iteratively and selectively until the optimal solution is found.

Note that feasibility is guaranteed by construction. The quality of the optimal solution for the reduced model following the second step should be quite good, as several variables with low  $d_{ij}$  values are already included in

the model. The key to a successful implementation of the last step is the identification of variables that will improve the objective function. It turns out that the simplex method of linear programming provides a basis to implement this step. The underlying theory is explained in the next section, and is subsequently applied to the file merge problem in Section 19.5.

As already highlighted, in practical applications the number of distance coefficients  $d_{ij}$ 's may be so large that it is impractical to store them. However, the coefficients do not need to be stored since it is possible to calculate  $d_{ij}$  from its definition during runtime. The value of all  $d_{ij}$ 's can be calculated from just  $|I| + |J|$  records. Clearly, calculating  $d_{ij}$  will consume significant runtime and therefore care should be given when specifying the expression to reduce calculation overhead. In AIMMS it is possible to specify the objective coefficients (like all parameters) using expressions. Consequently, the solution method presented above can be implemented in AIMMS such that the distance coefficients are calculated during runtime as required.

*Calculating  $d_{ij}$*

---

## 19.4 The simplex method

This section describes the simplex algorithm using matrix-vector notation for the underlying linear algebra. The algorithm forms the basis for the column evaluation technique used in this chapter, and the column generation technique used in Chapters 20 and 21.

*This section*

Without loss of generality all linear programming constraints can be written as equalities. Specifically, an inequality can be transformed to an equality by introducing a slack or surplus variable. In the Simplex method, the variables in the model are partitioned into two groups: the basic variables  $x_B$  and the nonbasic variables  $x_N$ . By definition, nonbasic variables are at one of their bounds (upper or lower) while basic variables are between their bounds. The matrices associated with the basic and nonbasic variables are denoted with  $B$  and  $N$ , respectively.

*Basic and nonbasic variables*

It is important to note that the choice of  $x$  here follows standard notation and it is not related to the  $x_{ij}$  used in the file merge model. Similarly, the matrix  $N$  is not related to  $N_i$  or  $N_j$ . The scope of this notation is limited to the current section.

*Notation*

**Minimize:**

$$c_B^t x_B + c_N^t x_N$$

*Partitioned linear program*

**Subject to:**

$$\begin{aligned} Bx_B + Nx_N &= b \\ x_B, x_N &\geq 0 \end{aligned}$$

After rewriting the equality constraint, and using the fact that the optimal basis is invertible, the basic variables  $x_B$  can be written in terms of the nonbasic variables  $x_N$ .

$$x_B = B^{-1}b - B^{-1}N x_N \geq 0$$

*Solution  
rewritten*

Next, this expression for  $x_B$  is substituted in the objective function to obtain the following form.

$$c_B^t B^{-1}b + (c_N^t - c_B^t B^{-1}N)x_N$$

*Objective  
function  
rewritten*

Taking the vector derivative of the last expression with respect to  $b$ , gives  $\lambda^t = c_B^t B^{-1}$ . This defines the *shadow price* of the associated constraint. As explained in Chapter 4, it is the rate of change of the objective function for a unit increase in the right-hand side of the constraint.

*Shadow prices*

Similarly, taking the vector derivative with respect to  $x_N$  gives the term  $(c_N^t - c_B^t B^{-1}N) = (c_N^t - \lambda^t N)$ . This defines the *reduced cost* of a variable. The reduced cost of a variable gives the rate of change of the objective function for a one unit increase in the bound of the variable. As discussed in Chapter 4, the reduced cost of a basic variable is zero. Reduced costs can be considered to be the sensitivity of the objective function value with respect to bounds associated with the nonbasic variables.

*Reduced costs*

As previously discussed, nonbasic variables  $x_N$  in the simplex method are at one of their bounds. During a simplex iteration, one of these variables is introduced into the basis, and a basic variable leaves the basis to become nonbasic. For the case, as in the file merge problem, where all variables are positive and nonbasic variables are at their lower bound, such an exchange is only of interest (for a minimization problem) when the corresponding component of the reduced cost vector  $(c_N^t - \lambda^t N)$  is negative. In this particular case, the objective function value will decrease when the value of the corresponding component of  $x_N$  is increased (away from its lower bound of zero). As soon as all components of the reduced cost vector  $(c_N^t - \lambda^t N)$  are nonnegative, no improvement in the objective function value can be made, and the current basic solution  $x_B = B^{-1}b$  is optimal. Note that, by definition, the reduced costs associated with basic variables are always zero.

*Simplex  
iteration*

---

## 19.5 Algorithmic approach

This section describes an algorithmic approach to solve the overall file merge model as a sequence of smaller submodels. The construction of each submodel is based on evaluating the reduced cost values of all variables as given by the simplex method. The inter-record distance  $d_{ij}$  are computed during runtime, and the corresponding variable is either put into a candidate list or ignored.

*This section*

Assume that the file merge model has been solved for a subset  $S$  of the variables  $x_{ij}$ ,  $(i, j) \in S$ . The resulting vector of shadow prices  $\lambda$  can be partitioned into  $\lambda^s$  for the supply constraints and  $\lambda^d$  for the demand constraints with components  $\lambda_i^s$  and  $\lambda_j^d$  respectively. Consider a particular combination of records  $i$  and  $j$ ,  $(i, j) \notin S$ . After computing  $d_{ij}$  directly from the two records, the quantity  $d_{ij} - (\lambda_i^s + \lambda_j^d)$  can be evaluated. Whenever this quantity is negative (i.e. may lower the objective function), the corresponding variable  $x_{ij}$  is a candidate variable.

*Candidate variables*

The first step of the algorithm is to find an initial feasible solution using an heuristic approach. Specifically, the records in each file are sorted with respect to 'Gross Family Income'. Next, in each file the first record with a positive value of 'Number of Families' is found. These two records are then merged to form a new record. The 'Number of Families' of the new record is equal to the minimum of the 'Number of Families' associated with the two input file records. The 'Number of Families' associated with each input file are adjusted by subtracting the smallest value of the two. The process is repeated until all records in both files have been considered, and the total number of families has been divided over the new records. All pairs of originating records  $i$  and  $j$  considered in this process, result in a basic variable  $x_{ij} > 0$ .

*Initial solution in words*

In addition to the variables identified in the previous paragraph a further selection can be made on the basis of small  $d_{ij}$  values for each  $i$ . The number of such additional variables can be as large as you desire. A typical value is between 5 to 25 extra variables for each  $i$ . Experience has shown that such a set is quite a good selection, and that for this selection the solution, the objective function value and the shadow prices of the submodel are close to optimal for the much larger model with all  $|I| \times |J|$  variables.

*Additional selection of variables*

Once an initial optimal solution for the current selection of variables has been computed, the algorithm visits all  $|I| \times |J|$  pairs of records. During this process there is an active search for candidate variables which, together with the variables from the previous submodel, will determine the next submodel to be solved. The entire process is repeated until there are no new candidates after visiting all possible pairs of records.

*Overall solution in words*

The flowchart in Figure 19.2 presents the computational steps to determine the initial values of  $S$  and  $x_{ij}$ . The set  $S$  contains all pairs  $(i, j)$  for which the corresponding variable  $x_{ij}$  is greater than zero. The element parameters  $i^*$  and  $j^*$  refer to records. Eventually, the  $x_{ij}$  values satisfy the equality constraints, and form a basic solution. At most  $|I| + |J|$  values of  $x_{ij}$  will be positive, because each iteration the (remaining) number of families from at least one record is assigned. The symbol  $\wedge$  represents the logical AND.

*Flowchart initial solution*

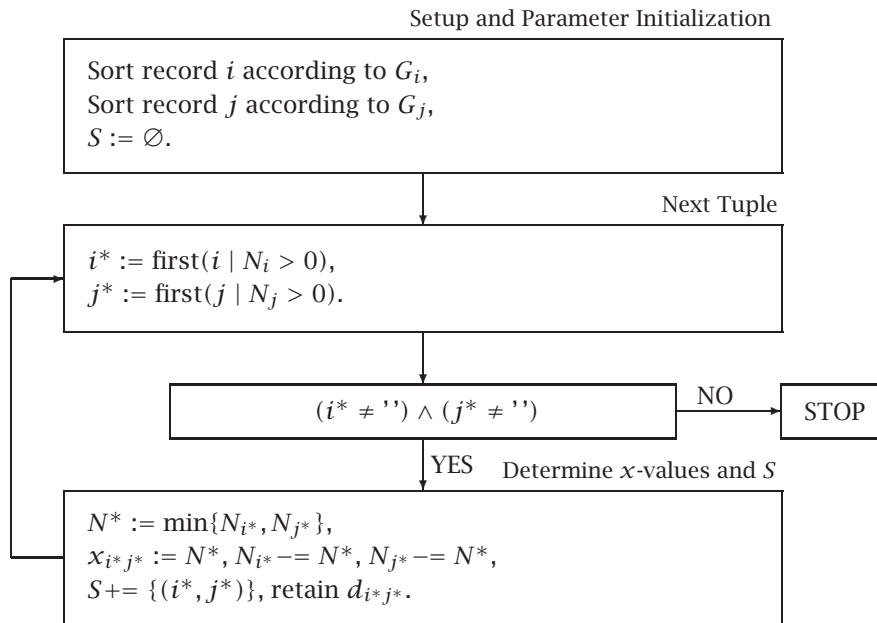


Figure 19.2: Flowchart initial solution

The flowchart in Figure 19.3 presents the computational steps to determine the optimal solution of the overall file merge model. Most of the notation has been introduced previously. New is that the element parameters  $i^*$  and  $j^*$  can be increased in value. That is, the assignment  $i^* += 1$  states that  $i^*$  refers to the next record in the sorted set  $I$  of records. Any reference beyond the last element is empty (i.e. ' ').

*Flowchart  
overall solution*

In the algorithm presented above there is no control over the size of the set  $S$  (the set of considered variables  $x_{ij}$ ). Control could be implemented by either deleting already considered variables or by limiting the number of candidate variables to be added. Deletion could be based on examining the (already considered) variables with the highest (instead of the lowest)  $d_{ij} - (\lambda_i^s + \lambda_j^d)$  value. Addition could be based on restricting the maximum number of candidates for each  $i$ .

*Computational  
considerations*

---

## 19.6 Summary

In this chapter the problem of merging of two files has been introduced as an application of the classical transportation problem. However, in practical applications the number of decision variable is extremely large and the resulting LP can not be solved in its entirety. To overcome this problem, a customized solution algorithm has been proposed. The proposal consists of a heuristic

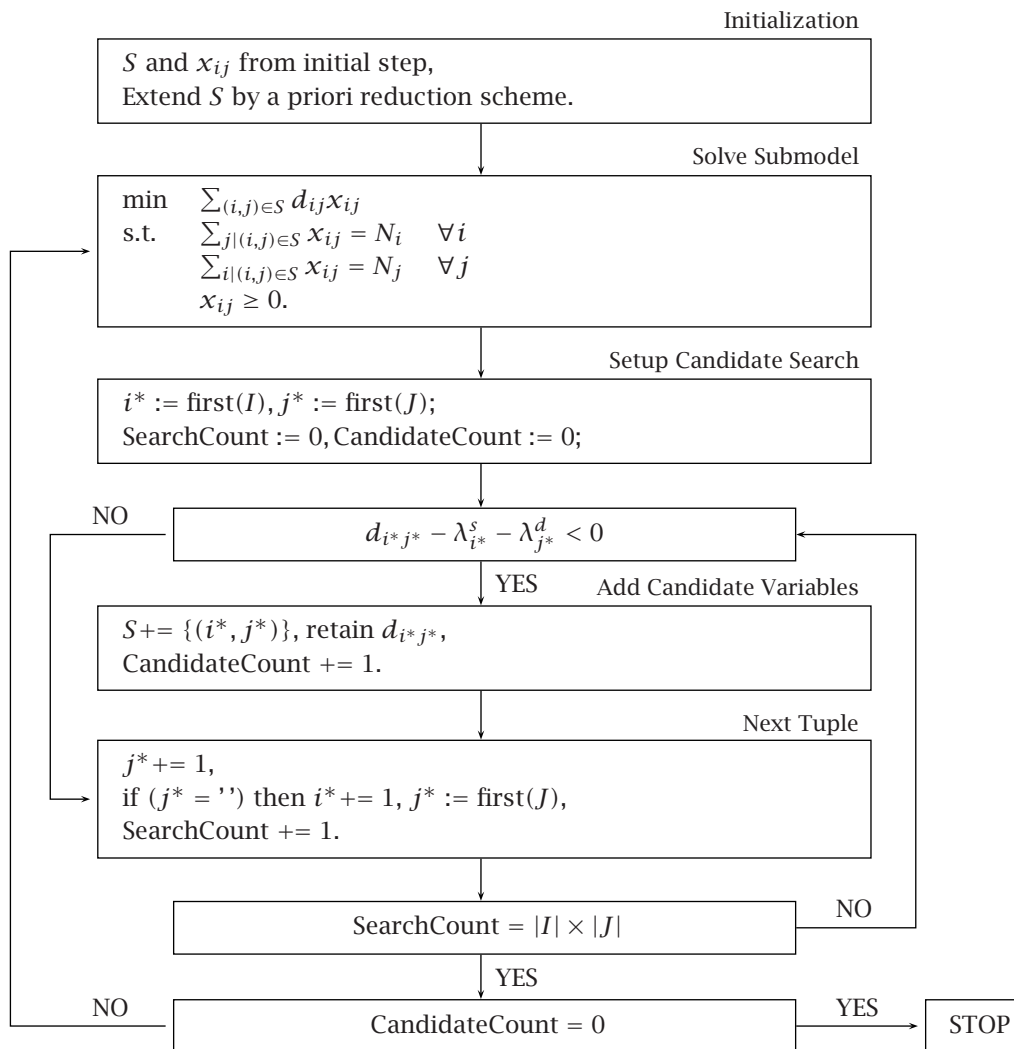


Figure 19.3: Flowchart algorithmic approach

approach to find initial solution values and shadow prices, followed by an algorithm to find the optimal solution of the model through systematically solving a sequence of smaller submodels. The underlying theory and detailed flowcharts have been presented.

## Exercises

- 19.1 Implement the file merge model presented in Section 19.2 using the first five records of the Income Data file and the Population Data File contained in Tables 19.1 and 19.2. Verify for yourself whether the

optimal solution found with AIMMS is the same as the one presented in Table 19.3.

- 19.2 How would you adjust your formulation of the model if the number of families in the Income Data File of Table 19.1 were 5 less for each of the five records?
- 19.3 Implement the algorithmic approach presented in Section 19.5 for the model and data referred to in the first exercise. Verify for yourself whether the optimal solution found is the same as the one found previously.

## Bibliography

- [Ch83] V. Chvátal, *Linear programming*, W.H. Freeman and Company, New York, 1983.
- [Gl92] F. Glover, D. Klingman, and N.V. Phillips, *Network models in optimization and their applications in practice*, John Wiley & Sons, New York, 1992.