BUREAU OF THE CENSUS
STATISTICAL RESEARCH DIVISION
Statistical Research Report Series
No. RR98/06

Set Covering Algorithms in Edit Generation

Bor-Chung Chen

U. S. Bureau of the Census
Statistical Research Division
Washington D.C. 20233

Report Issued: 9/1/98

Set Covering Algorithms in Edit Generation*

Bor-Chung Chen
Bureau of the Census
SRD, Rm. 3000-4, Washington, DC 20233-9100

September 1, 1998

Abstract

Results are presented from a comparison study of several set covering algorithms (routines) used in the implicit edit generation algorithms of Garfinkel, Kunnathur, and Liepins [1986] and Winkler [1997]. The edit generation algorithms are based on the Fellegi and Holt model [1976] of editing. Since the set covering routine is called many times in edit generation, an efficient routine will significantly reduce the computing time of the generation process. Unlike most of the applications of the set covering problem (SCP), in which an optimal cover is desirable, the edit generation is interested in finding all the prime covers to a SCP.

KEY WORDS: Explicit Edits, Redundant Covers, Subcovers, Integer Programming, Optimization

1. Introduction

The information gathered in any survey may contain inconsistent or incorrect data. These erroneous data need to be revised prior to data tabulations and retrieval. The revisions of the erroneous data should not affect the statistical inferences of the data. One of the important steps of this systematic revision process is computer editing. Fellegi and Holt [1976] provided the underlying basis of developing a computer editing system. An edit-generation algorithm, called the EGE algorithm, for the DISCRETE edit system was described in Winkler [1997]. The EGE algorithm is a much faster alternative to Algorithm 1, called the GKL algorithm, of Garfinkel, Kunnathur, and Liepins [1986]. In both of the EGE and GKL algorithms, the set covering

*This paper reports the results of research and analysis undertaken by Census Bureau staff. It has undergone a more limited review than official Census Bureau publications. This report is released to inform interested parties of research and to encourage discussion. The author thanks Dr. William E. Winkler for providing the FORTRAN code of the DISCRETE edit system and the examples. He is especially thankful to him for many helpful conversations in the detailed algorithm of the DISCRETE edit system.

routine is invoked many times to generate new implicit edits. Therefore, an efficient algorithm for the set covering problem becomes highly desirable to reduce the computation time of the edit generation. In this paper, a new set covering algorithm for the edit generation is described. This new algorithm significantly reduces the edit generation computation in several ways: (1) it does not produce redundant covers, and therefore avoids generating redundant edits in the edit generation process; (2) it minimizes the matrix size required to produce prime covers; (3) it makes the matrix reduction more effective; (4) it uses bitwise operations for more efficient computations.

The SCP in the DISCRETE edit system is applied twice. The first application is to find the minimal set of fields (the optimal solution) of a failed record to be modified to satisfy all explicit and implicit edits. The SCP is invoked once for each failed record. The second application is to find all the minimal sets of edits that are unioned to cover all possible values of a field, called a generating field. Each minimal set is used to generate an essentially new implicit edit as defined in section 2. The second application is NOT to find an optimal solution but to find all prime cover solutions to the SCP.

We examine a new set covering algorithm for generating essentially new implicit edits from a set of explicit edits in this paper. The explicit edits are initially specified by the subject matter experts for a particular survey.

2. Background

Let $\bar{E} = \{E^1, E^2, \cdots, E^m\}$ be a set of edits failed by a record $\boldsymbol{y}$ with n fields, consider the set covering problem:

$$\text{Minimize} \qquad \textstyle\sum_{j=1}^{n} c_j x_j$$

$$\text{subject to} \qquad \textstyle\sum_{j=1}^{n} a_{ij} x_j \geq 1, \qquad i = 1, 2, \cdots, m \tag{1}$$

$$x_j = \begin{cases} 1, & \text{if field } j \text{ is to be changed;} \\ 0, & \text{otherwise,} \end{cases}$$

where

$$a_{ij} = \begin{cases} 1, & \text{if field } j \text{ enters } E^i; \\ 0, & \text{otherwise,} \end{cases}$$

and c_j is a measure of "confidence" in field j. We need to get $\bar{E}$ from a *complete* set of edits to obtain a meaningful solution to (1). A complete set of edits is the set of explicit (initially specified) edits and all essentially new implied edits derived from them.

Notation: $\boldsymbol{a}= (a_1, a_2, \ldots, a_n)$ has n fields. $a_i \in A_i$ for each i $1 \leq i \leq n$, where A_i is the set of possible values or code values which may be recorded in

Field i. $|A_i| = n_i$. If $a_i \in A_i^o \subset A_i$, we also say

$$\boldsymbol{a} \in \boldsymbol{A}_i^o = A_1 \times A_2 \times \ldots \times A_{i-1} \times A_i^o \times A_{i+1} \times \ldots \times A_n.$$

The code space is $A_1 \times A_2 \times \ldots \times A_n = \boldsymbol{A}$.

Lemma 1 (Fellegi and Holt [1976]): If E^r are edits $\forall\ r \in S$, where S is any index set,

$$E^r : \bigcap_{j=1}^{n} \boldsymbol{A}_j^r = F, \quad \forall \;\; r \in S.$$

Then, for each i $(1 \leq i \leq n)$, the expression

$$E^* : \bigcap_{j=1}^{n} \boldsymbol{A}_j^* = F \tag{2}$$

is an implied edit, where

$$\boldsymbol{A}_j^* = \bigcap_{r \in S} \boldsymbol{A}_j^r \neq \emptyset \quad j = 1, \cdots, i-1, i+1, \cdots, n$$

$$\boldsymbol{A}_i^* = \bigcup_{r \in S} \boldsymbol{A}_i^r \neq \emptyset.$$

If all the sets A_i^r are proper subsets of A_i, i.e., $A_i^r \neq A_i$ (field i is an entering field of edit E^r) $\forall\ r \in S$, but $A_i^* = A_i$, then the implied edit (2) is called an *essentially new edit.* Field i, which has n_i possible values, is referred to as the *generating field* of the implied edit.

Let $\{E^r \mid r \in S\}$ be the set of the s edits with field i entering, then the set covering problem related to the generating field i is

$$\begin{aligned} &\text{Minimize} \quad \textstyle\sum_{r \in S} x_r \\ &\text{subject to} \quad \textstyle\sum_{r \in S} g_{rj}^i x_r \geq 1, \qquad j = 1, 2, \cdots, n_i \\ &\qquad\qquad x_r = \begin{cases} 1, & \text{if } E^r \text{ is in the cover;} \\ 0, & \text{otherwise,} \end{cases} \\ &\qquad\qquad r \in S \end{aligned} \tag{3}$$

where

$$g_{rj}^i = \begin{cases} 1, & \text{if } E^r \text{ contains the } j\text{th element in field } i; \\ 0, & \text{otherwise,} \end{cases}$$

is the jth element in field i of edit E^r $(r \in S)$. If $\boldsymbol{x}$ is a prime cover solution to (3) and $K = \{r \mid x_r = 1\} \subset S$, then $\cup_{k \in K} A_i^k = A_i$. A prime cover solution is a nonredundant set of the edits whose ith components cover all possible values of the entering field, which is the generating field to yield an essentially new implicit edit. This paper will concentrate on the algorithms that find all the prime cover solutions to the SCP (3), which is also referred to as a SCP with constraint matrix $\boldsymbol{G}=(g^i_{rj})_{s\times n_i}$. We will also compare the performance of the set covering algorithm, referred to as the old algorithm, implemented in the DISCRETE edit system and the new algorithm described in this paper.

Overview of the old algorithm Given $\boldsymbol{G}=(g^i_{rj})_{s\times n_i}$ and its rth row and jth column vectors are $\boldsymbol{g}_{r\cdot}=[g^i_{r1},g^i_{r2},\cdots,g^i_{rn_i}]$ and $\boldsymbol{g}_{\cdot j}=[g^i_{1j},g^i_{2j},\cdots,g^i_{sj}]^T$, respectively, then the old algorithm is described as following:

1. Search for unit column vectors from $\boldsymbol{g}_{\cdot j}$, $1\leq j\leq n_i$. Let the unit column vectors obtained are $\boldsymbol{I}^{j_1}_{r_1}$, $\boldsymbol{I}^{j_2}_{r_2}$, $\cdots$, $\boldsymbol{I}^{j_k}_{r_k}$, where $\boldsymbol{I}^{j_1}_{r_1}=\boldsymbol{g}_{\cdot j_1}$ is a unit column vector with the r_1th element 1 and the other elements 0.

2. Assume that $r_1,r_2,\cdots,r_k$ are k different numbers. Remove rows $r_1,r_2,\cdots,$ r_k and the l columns with $g^i_{rj}=1$, where $r=r_1,r_2,\cdots,$or r_k and $j=j_1,j_2,\cdots,$or j_l from $\boldsymbol{G}$ to form a reduced matrix $\boldsymbol{G}_1$. Assume that $j_1,j_2,\cdots,j_l$ are l different numbers. Then $\boldsymbol{G}_1$ with dimension $(s-k)\times(n_i-l)$ is the constraint matrix to a newly reduced SCP, where $l\geq k$.

3. If $l=n_i$, the set of $\boldsymbol{g}_{r_1\cdot}$, $\boldsymbol{g}_{r_2\cdot}$, $\cdots$, $\boldsymbol{g}_{r_k\cdot}$ from $\boldsymbol{G}$ is the only prime cover solution, STOP; otherwise it is part of the cover solution(s) found in the following steps, CONTINUE.

4. Form an Edit Cover Forest (ECF), as illustrated in Figure 1 for an example with $(s-k)=4$ edits to the newly reduced SCP. Each node of the ECF represents a set of row vectors from $\boldsymbol{G}_1$, e.g., node 234 in Figure 1 represents the set of row vectors $\boldsymbol{g}_{2\cdot}$, $\boldsymbol{g}_{3\cdot}$, and $\boldsymbol{g}_{4\cdot}$ from $\boldsymbol{G}_1$.

5. Traverse the trees of the ECF in preorder. At each node, test the set of row vectors to determine if it is a cover solution to the newly reduced SCP. If it is, combine the set with the set from Step 3 and the combined set is a cover solution to the SCP (3).

The old algorithm is inefficient when $s\gg 2^{n_i}-2$. This is because there would be many different edits with $E^p\neq E^q$, but with $\boldsymbol{g}_{p\cdot}$=$\boldsymbol{g}_{q\cdot}$, which makes Step 1 of the algorithm ineffective. Also, the number of nodes to be visited in the ECF becomes large with small k and large s. Another inefficiency of the algorithm is that it finds all the prime and redundant covers. Redundant covers will generate redundant implicit edits in the edit generation algorithm.

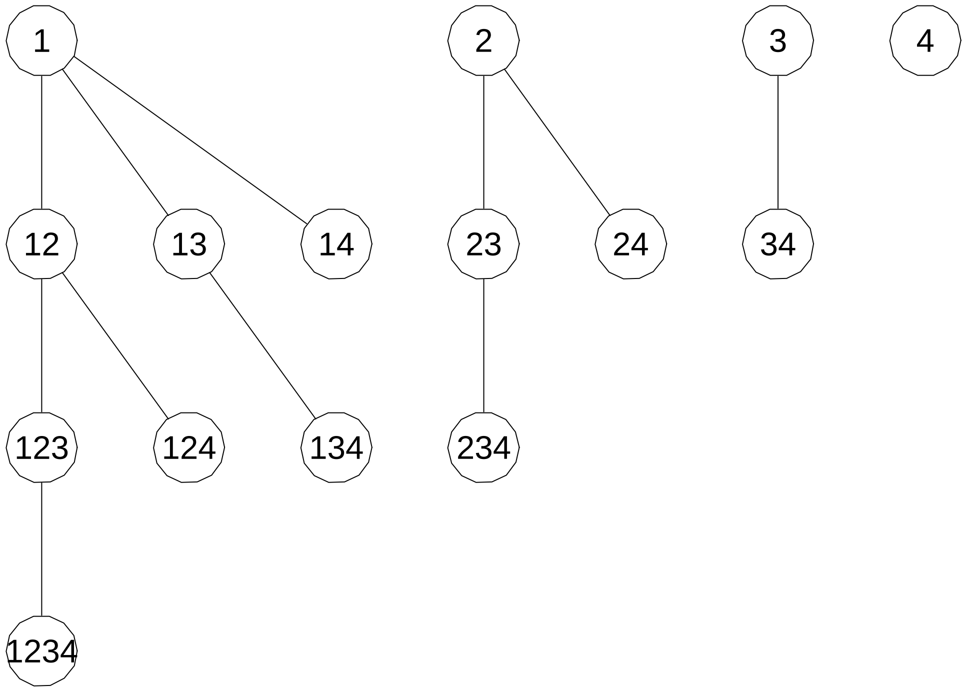

Figure 1. An Edit Cover Forest

3. New Set Covering Algorithm

The performance of the set covering algorithm used in edit generations can be improved if the size (the number of nodes) of the ECF is significantly reduced. Each removal of a row vector from $\boldsymbol{G}$ in (3) will reduce the size more than 50%. The following is the underlying theorem for removing duplicated row vectors from $\boldsymbol{G}$:

Theorem 1: In the SCP (3), if $\boldsymbol{g}_{p\cdot}= \boldsymbol{g}_{q\cdot}$ and $\{\boldsymbol{g}_{r_1\cdot}, \cdots, \boldsymbol{g}_{r_k\cdot}, \boldsymbol{g}_{p\cdot}\}$ is a prime cover solution to the following reduced SCP:

$$\begin{array}{rl} \text{Minimize} & \sum_{r\in S-\{q\}} x_r \\ \text{subject to} & \sum_{r\in S-\{q\}} g^i_{rj} x_r \geq 1, \qquad j = 1, 2, \cdots, n_i \end{array} \tag{4}$$

then both of $\{\boldsymbol{g}_{r_1\cdot}, \cdots, \boldsymbol{g}_{r_k\cdot}, \boldsymbol{g}_{p\cdot}\}$ and $\{\boldsymbol{g}_{r_1\cdot}, \cdots, \boldsymbol{g}_{r_k\cdot}, \boldsymbol{g}_{q\cdot}\}$ are prime cover solutions to the SCP (3).

Although the maximum number of different row vectors is $2^{n_i} - 2$, the actual size is much smaller due to the limited number of edit patterns for each field in $\{E^r \mid r \in S\}$. When $\boldsymbol{g}_{p\cdot}= \boldsymbol{g}_{q\cdot}$, it means that the pth and qth edits in $\{E^r \mid r \in S\}$ take exactly the same set of values for the entering field i.

Example (Fellegi and Holt [1976]): If a questionnaire contains three fields:

Field Number i	Field Name	Possible Codes	A_i
1	Age	0–14(1), 15+(2)	{1, 2}
2	Marital Status	Single(1),Married(2) Divorced(3),Widowed(4) Separated(5)	{1,2,3,4,5}
3	Relationship to Head of Household	Head(1), Spouse of Head(2), Other(3)	{1,2,3}

and there are three edits:

Edit i	A_1^i	A_2^i	A_3^i
1	{1}	{2, 3, 4, 5}	{1, 2, 3}
2	{1, 2}	{1, 3, 4, 5}	{2}
3	{1}	{1, 2, 3, 4, 5}	{2}

Edits 1 and 3 (E^1 and E^3) are two different edits with entering field 1, but $\boldsymbol{g}_{1\cdot} = \boldsymbol{g}_{3\cdot} = [1, 0]$. In this example, the SCP (3) has $s = 2$ but has only one distinct row vector of $\boldsymbol{g}_{r\cdot}$.

Lemma 2 (Garfinkel and Nemhauser [1972]): If $\boldsymbol{g}_{\cdot u} \geq \boldsymbol{g}_{\cdot v}$ (i.e., $g_{ru}^i \geq g_{rv}^i$ $\forall$ $r \in S$) for some u and v, then every cover of column v covers column u.

The objective of Theorem 1 and Lemma 2 is to reduce the matrix size of $\boldsymbol{G}$ and therefore minimize the size of the ECF before an actual set covering routine is invoked. A new set covering routine is described below to minimize the number of nodes to be visited in the ECF. The new routine is not interested in any cover other than the prime covers. As in the old set covering algorithm, the ECF is traversed in preorder.

Figure 1 shows that if a node is found to be a cover, its offspring nodes are not prime covers and will not be visited at all. Therefore, unlike the old algorithm, we would like to find a cover node as close as possible to any of the root nodes. This will minimize the number of nodes needed to be visited and the computation to find a prime cover. For example, if node 12 is found to be a cover, nodes 123, 1234, and 124 will be skipped because they won't be a prime cover. The next node to be visited will be node 13. At each cover node, redundant row vectors will be removed to form a prime cover. If the number of redundant row vectors is 0 or 1, no additional computation is needed to find a prime cover. If it is more than 1, new covers are obtained by removing the redundant row vectors one at a time. This same procedure of removing redundant row vectors is applied to each of the new covers.

At each cover node, if a subcover of the cover node is found to be a cover and the node representing the subcover is not yet visited, then the node and its offspring nodes are marked visited. For example, if the current node 123 in Figure 1 is found to be a cover and 23 is a subcover, then nodes 23 and 234 are marked visited. Also, the procedure just described will not miss any prime covers. If we use the same example, node 1234 will not be visited because the current node 123 is a cover. If 34 is a subcover of the node 1234, it will be visited at node 34 unless it is marked visited before it is reached in preorder.

As described above, we would like to find a cover node in the ECF as close as possible to any of the root nodes. This can be accomplished by performing an additional step before the ECF is formed. The row vectors, $\boldsymbol{g}_{r\cdot}$, are sorted based on the number of 1's in descending order. The following example provides information about the computation saved at the expense of sorting.

Example Suppose a reduced matrix $\boldsymbol{G}_1$ is

$$\boldsymbol{G}_1 = \begin{pmatrix} 0&0&1&0\\ 1&1&0&1\\ 1&0&1&1\\ 0&1&1&0\\ 0&1&1&0\\ 0&0&0&1\\ 1&1&0&0\\ 0&1&1&1 \end{pmatrix} \quad \text{and} \quad \boldsymbol{G}_1^a = \begin{pmatrix} 0&0&1&0\\ 0&0&0&1\\ 0&1&1&0\\ 0&1&1&0\\ 1&1&0&0\\ 1&1&0&1\\ 1&0&1&1\\ 0&1&1&1 \end{pmatrix} \boldsymbol{G}_1^d = \begin{pmatrix} 1&1&0&1\\ 1&0&1&1\\ 0&1&1&1\\ 0&1&1&0\\ 0&1&1&0\\ 1&1&0&0\\ 0&0&1&0\\ 0&0&0&1 \end{pmatrix}$$

are two matrices sorted with $\boldsymbol{g}_{r\cdot}$ based on the number of 1's in ascending and descending order, respectively. The number of nodes of the ECF for the three matrices is $2^8 - 1 = 255$. The following table shows the number of nodes visited and the number of nodes visited which have the number of redundant row vectors less than 2 for the three matrices:

Matrix	$\boldsymbol{G}_1^a$	$\boldsymbol{G}_1$	$\boldsymbol{G}_1^d$
Number of nodes in ECF	255	255	255
Number of nodes visited	70	36	21
Number of nodes visited with <2 redundant row vectors	33	24	16
Percentage	47.1%	66.7%	76.2%

The table also shows the percentage of the cover nodes visited with less than two redundant row vectors. This indicates that the computing time is significantly reduced with $\boldsymbol{G}_1^d$, a sorted matrix of the row vectors based on the number of 1's in descending order.

Description of the new algorithm The new algorithm is designed for reducing the number of row vectors needed to find prime covers for the SCP (3).

The reduction of the row vectors in $\boldsymbol{G}$ will reduce the size of the corresponding ECF formed by the row vectors. The following is a step by step description of the new algorithm:

1. Use the quick sorting algorithm to remove the duplicate row vectors from the matrix $\boldsymbol{G}$ in the SCP (3). Let $\boldsymbol{H}= (h^i_{rj})_{t\times n_i}$ be the reduced matrix, in which no two row vectors are equal, where $t \leq s$ and the rth row and jth column vectors are $\boldsymbol{h}_{r\cdot} = [h^i_{r1}, h^i_{r2}, \cdots, h^i_{rn_i}]$ and $\boldsymbol{h}_{\cdot j} = [h^i_{1j}, h^i_{2j}, \cdots, h^i_{tj}]^T$, respectively.

2. (Lemma 2) For each pair (u, v), $1 \leq u \neq v \leq n_i$, test if $\boldsymbol{h}_{\cdot u} \geq \boldsymbol{h}_{\cdot v}$. If yes, remove the column vector $\boldsymbol{h}_{\cdot u}$ from $\boldsymbol{H}$. Let $\boldsymbol{B}= (b^i_{rj})_{t\times m_i}$ be the further reduced matrix, where $m_i \leq n_i$ and the rth row and jth column vectors are $\boldsymbol{b}_{r\cdot} = [b^i_{r1}, b^i_{r2}, \cdots, b^i_{rm_i}]$ and $\boldsymbol{b}_{\cdot j} = [b^i_{1j}, b^i_{2j}, \cdots, b^i_{tj}]^T$, respectively.

3. Search for unit column vectors from $\boldsymbol{b}_{\cdot j}$, $1 \leq j \leq m_i$. Let the unit column vectors obtained are $\boldsymbol{I}^{j_1}_{r_1}$, $\boldsymbol{I}^{j_2}_{r_2}$, $\cdots$, $\boldsymbol{I}^{j_k}_{r_k}$, where $\boldsymbol{I}^{j_1}_{r_1} = \boldsymbol{b}_{\cdot j_1}$ is a unit column vector with the r_1th element 1 and the other elements 0.

4. Assume that $r_1, r_2, \cdots, r_k$ are k different numbers. Remove rows $r_1, r_2, \cdots$, r_k and the l columns with $b^i_{rj} = 1$, where $r = r_1, r_2, \cdots,$or r_k and $j = j_1, j_2, \cdots,$or j_l from $\boldsymbol{B}$ to form a reduced matrix $\boldsymbol{B}_1$. Assume that $j_1, j_2, \cdots, j_l$ are l different numbers. Then $\boldsymbol{B}_1$ with dimension $(t - k) \times (m_i - l)$ is the constraint matrix to a newly reduced SCP, where $l \geq k$.

5. If $l = m_i$, the set of $\boldsymbol{b}_{r_1\cdot}$, $\boldsymbol{b}_{r_2\cdot}$, $\cdots$, $\boldsymbol{b}_{r_k\cdot}$ from $\boldsymbol{B}$ is the only prime cover solution to the reduced SCP with the constraint matrix $\boldsymbol{B}$, GO TO Step 12; otherwise it is part of the cover solution(s) found in the following steps, CONTINUE.

6. Use the quick sorting algorithm to rearrange the row vectors of $\boldsymbol{B}_1$ based on the number of 1's in the row vectors in descending order. Let $\boldsymbol{D}= (d^i_{rj})_{(t-k)\times(m_i-l)}$ be the new matrix and the rth row and jth column vectors are $\boldsymbol{d}_{r\cdot} = [d^i_{r1}, d^i_{r2}, \cdots, d^i_{r(m_i-l)}]$ and $\boldsymbol{d}_{\cdot j} = [d^i_{1j}, d^i_{2j}, \cdots, d^i_{(t-k)j}]^T$, respectively.

7. Form an ECF, as illustrated in Figure 1 for an example with $(t - k) = 4$ edits to the SCP with constraint matrix $\boldsymbol{D}$. Each node of the ECF represents a set of row vectors from $\boldsymbol{D}$, e.g., node 234 in Figure 1 represents the set of row vectors $\boldsymbol{d}_{2\cdot}$, $\boldsymbol{d}_{3\cdot}$, and $\boldsymbol{d}_{4\cdot}$ from $\boldsymbol{D}$.

8. Traverse the trees of the ECF in preorder. At each unvisited node, test the set of row vectors to determine if it is a cover solution to the newly reduced SCP. If it is, GO TO Step 9.

9. Identify the redundant row vectors of the cover solution. If the number of the redundant row vectors is less than 2, the set of the nonredundant row vectors is a prime cover solution. Otherwise new subcovers are obtained by removing a redundant row vector one at a time and GO TO Step 9 for each new subcover. If there is an unvisited node in the ECF corresponding to one of the subcovers, mark the node and its offspring as visited.

10. Skip the offspring nodes of the cover node just identified and GO TO Step 8.

11. All the prime cover solutions, if any, to the SCP with constraint matrix $\boldsymbol{D}$ are found. Trace back to the original indices in $\boldsymbol{B}$ of the row vectors in the prime cover solutions in $\boldsymbol{D}$ and combine with $\boldsymbol{b}_{r_2\cdot}, \cdots, \boldsymbol{b}_{r_k\cdot}$ in Step 5 to form prime cover solutions to the SCP with constraint matrix $\boldsymbol{B}$.

12. For each prime cover solution to the SCP with constraint matrix $\boldsymbol{B}$, trace back to the original indices in $\boldsymbol{G}$ of the row vectors in the cover solution. If any row vector has duplicates, replace the row vector with each duplicate to form a new prime cover solution to the SCP with constraint matrix $\boldsymbol{G}$.

4. Implementation of the Algorithms and their Performance

The old set covering algorithm in the DISCRETE system was written in FORTRAN 77. The new algorithm is implemented with C++ and bitwise operations are used to find the cover solutions.

The intent of the new algorithm is for the application of the SCP used in edit generation. Therefore, the computational experience discussed in this section is based on edit generation examples. The computations were performed on a Sun UltraSparc. The first example has 33 fields and 252 explicit edits. One of the 33 fields is not an entering field for all 252 explicit edits. The following table shows the dimensions of the constraint matrix $\boldsymbol{G}$ to the SCP (3) and the number of matrices (under the heading "#") for each dimension listed, which is also the number of entering fields that have a constraint matrix with that dimension.

dim($\boldsymbol{G}$)	#	dim($\boldsymbol{G}$)	#	dim($\boldsymbol{G}$)	#	dim($\boldsymbol{G}$)	#
1×12	1	11×2	1	16×2	2	30×2	2
40×12	1	12×2	1	17×2	3	31×2	1
44×12	1	13×2	2	25×2	4	55×2	1
2×2	2	14×2	2	26×2	1	70×2	1
8×2	2	15×2	3	27×2	1	-	-

For example, there is an entering field that has the constraint matrix with dimension 40×12. And it has 40 of the 252 explicit edits that has the field entering and 12 value states for the field. The input data were repeated 100, 200, and 300 times to have a better measurement in seconds. Table 1 shows the performance of the two algorithms and the new algorithm is about 166 times faster than the old algorithm.

Table 1. Performance of the Two Algorithms for Example 1

number of times input data repeated	old algorithm CPU time (seconds)	new algorithm CPU time (seconds)
100	83.0	0.5
200	166.0	1.0
300	250.0	1.5

The second example uses the decennial census long form questionnaire. It has 52 fields and 523 explicit edits. Six of the 52 fields are not entering fields for all 523 explicit edits. The following table shows the dimensions of the constraint matrix $\boldsymbol{G}$ to the SCP (3) and the number of matrices for each dimension listed.

dim($\boldsymbol{G}$)	#	dim($\boldsymbol{G}$)	#	dim($\boldsymbol{G}$)	#	dim($\boldsymbol{G}$)	#	dim($\boldsymbol{G}$)	#
1×14	1	28×3	1	34×3	3	39×3	3	58×3	1
29×14	2	30×3	1	35×3	3	40×3	2	80×3	1
1×3	10	31×3	1	36×3	2	41×3	1	84×3	1
3×3	2	32×3	1	37×3	2	42×3	1	88×3	1
27×3	1	33×3	2	38×3	1	47×3	1	91×3	1

Table 2 shows the performance of the two algorithms. The new algorithm is about 49 times faster than the old algorithm.

Table 2. Performance of the Two Algorithms for Example 2

number of times input data repeated	old algorithm CPU time (seconds)	new algorithm CPU time (seconds)
100	66.0	0.9
200	132.0	2.7
300	198.0	3.8

5. Discussion and Summary

Most of the fields have two value states in Example 1 and three in Example 2. However, the number of explicit edits which have the fields entering can be as many as 70 for Example 1 and 91 for Example 2. For instance, the constraint matrix $\boldsymbol{G}$ with dimension 91×3 in Example 2 has many duplicates of row vectors. The maximum number of different row vectors of the matrix is

$2^3 - 2 = 6$. The number of nodes in the ECF with the old algorithm is $2^{91} - 1$ and the maximum number of nodes with the new algorithm is $2^6 - 1 = 63$. The new algorithm requires additional computations in the order of $91 \log 91$ for sorting. Therefore, for a constraint matrix with dimension $m \times n$ the new algorithm is very efficient if $m \gg 2^n - 2$. Also, Step 1 of the old algorithm is not effective because there are so many duplicates of row vectors in the matrix $\boldsymbol{G}$ and it is almost impossible to locate a unit column vector when $m \gg 2^n - 2$. Another advantage of the new algorithm is that it finds the prime cover solutions only. The redundant cover solutions found with the old algorithm result in unnecessary computations of the edit generations in the DISCRETE edit system.

References

[1] I. P. Fellegi and D. Holt. A systematic approach to automatic edit and imputation. *Journal of the American Statistical Association*, 71:17–35, 1976.

[2] R. S. Garfinkel, A. S. Kunnathur, and G. E. Liepins. Optimal imputation of erroneous data: Categorical data, general edits. *Operations Research*, 34:744–751, 1986.

[3] R. S. Garfinkel and G. L. Nemhauser. *Integer Programming*. John Wiley & Sons, New York, 1972.

[4] W. E. Winkler. Set-covering and editing discrete data. Technical report, Bureau of the Census, 1997.