UNIVERSIDAD NACIONAL AUTONOMA DE MEXICO

FACULTAD DE CIENCIAS

METODOLOGIA DE FELLEGI Y HOLT: VALIDACION E IMPUTACION DE DATOS

T E S I S

C O N J U N T A

QUE PARA OBTENER EL TITULO DE:

ACTUARIA Y MATEMATICO

P R E S E N T A N :

MARGOT ALEJANDRA PALACIOS OSTRIA

EDGAR JAVIER GONZALEZ LICEAGA

DIRECTORES DE TESIS: MAT. MARGARITA ELVIRA CHAVEZ CANO
ACT. ERIC MANUEL RODRIGUEZ HERRERA

2004

UNIVERSIDAD NACIONAL AVENIDA DE MEXICO

ACT. MAURICIO AGUILAR GONZÁLEZ
Jefe de la División de Estudios Profesionales de la Facultad de Ciencias
Presente

Comunicamos a usted que hemos revisado el trabajo escrito:

"Metodología de Fellegi y Holt: Validación e Imputación de Datos"

realizado por Edgar Javier González Liceaga con número de cuenta 09957379-1 y Margot Alejandra Palacios Ostria con número de cuenta 09958346-4, quienes cubrieron los créditos de las carreras de Matemáticas y Actuaría, respectivamente.

Dicho trabajo cuenta con nuestro voto aprobatorio.

Atentamente

Director de Tesis Propietario	Mat. Margarita Elvira Chávez Cano	
Director de Tesis Propietario	Act. Eric Manuel Rodríguez Herrera	
Propietario	Dra. Guillermina Eslava Gómez	
Suplente	M. en C. Jesús Agustín Cano Garcés	
Suplente	Act. Jaime Vázquez Alamilla	

Consejo Departamental de Matemáticas

M. en C. Alejandro Bravo Mojica — Act. Jaime Vázquez Alamilla

FACULTAD DE CIENCIAS
CONSEJO DEPARTAMENTAL
DE
MATEMATICAS

AGRADECIMIENTOS

Agradecemos:

A nuestra asesora de tesis, la Mat. Margarita Elvira Chávez Cano, que, gracias a sus cursos, nos hizo interesarnos en el campo de la Estadística, inculcándonos siempre el trabajo de calidad. Gracias por orientarnos en nuestro proyecto de tesis y por estimularnos a seguir adelante a pesar de los obstáculos que se nos presentaron.

A nuestro asesor de tesis, el Act. Eric Manuel Rodríguez Herrera, por habernos hecho partícipes del proyecto que ha venido desarrollando en el INEGI. Gracias por introducirnos en el tema, por darnos las bases de nuestra tesis y por habernos dado la oportunidad de dar a conocer a otros el trabajo que realizamos.

A ambos por el gran apoyo y tiempo invertido en la realización de este trabajo. Sin ustedes hubiera sido imposible alcanzar nuestra meta.

Edgar y Ale.

Agradezco:

A Dios por permitirme llegar a este día, por permitir realizar mi sueño...

A mis padres por sus desvelos, por los grandes esfuerzos que han realizado para que yo sea quien soy, por su cariño y consejos y sobre todo por el gran apoyo y confianza que siempre me han brindado. Este logro es también suyo.

A mis hermanos por el gran apoyo que me brindaron a lo largo de mi carrera, por su cariño y comprensión. A Toño por ser mi ejemplo a seguir y a Nelly por ser mi hermana y amiga.

A Vale por su cariño y amor, por exaltarme a ser cada día mejor, por creer en mí. Eres y seguirás siendo alguien importante en mi vida.

A Edgar por su amistad, confianza y por haber compartido conmigo su tiempo tanto en la realización de esta tesis, así como en tantas otras actividades en las cuales siempre nos supimos complementar. Sin ti hubiera sido más difícil alcanzar este logro.

Ale.

Agradezco:

Ma mère pour son appui pendant toutes ces années de développement personnel; même à la distance je t'ai toujours senti à côté de moi.

Mon père pour ce moment d'appui, quand j'en ai eu tant besoin.

A Ale, por haber trabajado juntos a lo largo de la realización de esta tesis y de gran parte de la carrera. Gracias a tu impulso este trabajo se vio terminado.

My friends, Abraham, Claudia, José Luis and Fabrizzio for all we've passed through during these last years and for their guidance on this transition toward Biology.

A te, Bibiano, per tutto quello che mi hai dàto in questi quasi tre anni insieme. Sài quanto rappresenti per me.

Edgar.

ÍNDICE

Página

Página

[1] Con base a las ideas dadas en I. P. Fellegi y D. Holt (1976).
[2] Con base en los algoritmos dados en Robert S. Garfinkel *et al.* (1986).
[3] Con base en el algoritmo dado en William E. Winkler (1997).
[4] Con base en el algoritmo dado en Bor-Chung Chen (1998).
[5] Con base en los algoritmos dados en William E. Winkler y Bor-Chung Chen (2002).

INTRODUCCIÓN

Las bases de datos generadas por censos o encuestas pueden contener grandes cantidades de información de la cual una proporción significativa es incorrecta. Los errores surgen debido a que las preguntas son ambiguas, fueron mal contestadas (voluntaria o involuntariamente) o debido a errores en la trascripción o codificación de las mismas. Estos errores deben ser detectados y corregidos para obtener bases de datos confiables. Ambas actividades, denominadas validación e imputación respectivamente, son fundamentales dentro del proceso de análisis de una encuesta. Iniciamos nuestro trabajo definiendo, en las siguientes secciones, estas dos actividades.

I.1 VALIDACIÓN.

Por ***validación*** entendemos las siguientes acciones:

A. La revisión de cada uno de los campos de la encuesta para asegurar que contenga una entrada válida.

   Esta revisión implica:

   - Determinar si una entrada en blanco es válida, es decir, si es un blanco por pase.

- Determinar si el valor de una entrada está dentro del conjunto de valores válidos para el campo.

Esta validación se considera como consecuencia inmediata del cuestionario y de la estructura de la entrada.

Ejemplo I.1:

Supongamos que en una encuesta, el cuestionario incluye los siguientes campos:

Sexo = {{Masculino}, {Femenino}}
Edad = {{0}, {1}, {2}, ..., {100+}}
Estado civil = {{Soltero}, {Casado}, {Divorciado}, {Viudo}, {Separado}}
Número de hijos = {{0}, {1}, {2}, ..., {20+}}.

Así, la respuesta a cada campo por parte del encuestado deberá tomar alguno de estos valores.

B. La comprobación de consistencia en ciertas combinaciones de campos predeterminados.

Esta comprobación involucra:

- Especificar conjuntos de valores que no son aceptables juntos para combinaciones específicas de campos en un registro.

Ejemplo I.2:

Edad = {{0}, {1}, {2}, ..., {10}} implica Estado civil = {Soltero}.

C. La comprobación de consistencia entre ciertas combinaciones de registros.

Esta comprobación involucra:

- Especificar conjuntos de valores que no son aceptables juntos para combinaciones específicas de campos en diferentes registros.

Ejemplo I.3:

Supongamos que en un hogar encuestado hay tres integrantes, como por ejemplo un jefe de familia, un cónyuge y la madre del jefe de familia. Entonces tendríamos que:

Edad de la madre - Edad del jefe de familia □ 15.

I.2 IMPUTACIÓN.

Por ***imputación*** entenderemos la asignación de nuevos valores a aquellos campos, en uno o más registros, que necesitan ser corregidos para pasar el proceso de validación.

Ejemplo I.4:

> Supongamos que tenemos en un registro un individuo que declara tener ocho años y estar casado. En este caso el registro falla la regla de validación dada en el Ejemplo I.2 por lo que se requiere imputar dicho registro. Una posible imputación sería asignar una nueva edad al individuo; por ejemplo, 28 años.

I.3 IMPORTANCIA DE LA VALIDACIÓN Y LA IMPUTACIÓN.

El aceptar todos los datos tal como vienen, además de generar problemas en la presentación de los resultados, puede causar también graves complicaciones en el proceso, sobre todo si éste se hace de manera automática. Por ello un nivel mínimo de validación e imputación siempre es conveniente.

Los argumentos que habitualmente se citan a favor de la validación e imputación de los datos son:

1. Se obtienen beneficios para las siguientes fases del procesamiento de los datos. Dichas fases se verían afectadas por la aparición de valores fuera del conjunto de los admisibles, variables sin valor asignado o inconsistencias entre los valores de las variables.

2. De cara al usuario los datos ganan en credibilidad. Así, una tabla cruzando edades y estado civil, en el que sobre una población de 2 millones de individuos apareciesen dos viudas menores de 10 años, será poco fiable, sobre todo para el lector profano, que no está al tanto de la complejidad de las tareas necesarias para elaborar la tabla. Si se eliminasen por cualquier procedimiento esos dos casos de viudas precoces, el lector vería la tabla con mayor confianza.

3. En muchas ocasiones los usuarios hacen caso omiso de las rúbricas de las tablas correspondientes a la no respuesta, considerando que las demás rúbricas corresponden a la imagen real de la población estudiada. Al comportarse así están suponiendo que los no encuestados se comportan igual que el resto de la población. Esto es una especie de imputación pero a un nivel muy básico. Como señalan I. P. Fellegi y D. Holt (1976), parece más razonable hacer la imputación a nivel de cada registro, aprovechando la redundancia interna de éste para imputar las variables sin respuesta.

4. También se afirma la conveniencia de que sea el organismo central de estadística, que ha recogido los datos, el que los ofrezca validados e imputados, en lugar de que sea cada usuario el que haga ajustes. De esta manera se gana en uniformidad y

comparabilidad de los resultados de los análisis llevados a cabo por los distintos usuarios.

5. Los casos de sustitución de un valor en un registro erróneo por otro correcto se defienden con el argumento de que en ocasiones es la solución viable más rápida, en contraposición a otras como sería la de recontactar al encuestado. Además, la calidad de los datos no se ve afectada en sentido negativo con estas imputaciones.

I.4 COSTO DE LA VALIDACIÓN Y LA IMPUTACIÓN.

De acuerdo al Subcomité sobre validación de datos en agencias estadísticas federales (1990) de Estados Unidos, alrededor de tres cuartas partes de las encuestas realizadas por dichas agencias tienen costos de validación e imputación que representan al menos 20% del costo total de la encuesta. La mediana se encuentra en el 35%.

Los costos de validación e imputación como un porcentaje del costo total de la encuesta varían grandemente con el tipo de encuesta. Las encuestas demográficas tienen una mediana del 20% comparada con el 40% de las económicas. Entre las encuestas económicas, aquellas que usan registros administrativos tienen la mediana más alta, 60%. Este alto porcentaje no necesariamente indica un alto costo absoluto de validación e imputación sino que podría indicar un bajo costo total de la encuesta ya que no se recolectan nuevos datos.

Las encuestas en las que toda la corrección de errores es hecha por oficinistas o analistas tienen costos de validación e imputación que representan más del 40% del costo total de la encuesta comparado con aquellas en las que sólo casos inusuales son referidos a los analistas.

Para el caso de los censos y encuestas realizados por el Instituto Nacional de Estadística, Geografía e Informática (INEGI), los costos son muy variados debido a que los sistemas de validación e imputación son desarrollados para cada censo o encuesta en forma específica. Así encontramos que el Censo de Población y Vivienda y los Censos Económicos cuentan con procesos automatizados, mientras que las encuestas económicas realizan los procesos de validación e imputación de forma manual, lo que genera grandes diferencias en los costos. Por ejemplo, en el Censo de Población y Vivienda se estima que un 10% del costo total del Censo corresponde a la validación y la imputación; en las encuestas económicas el porcentaje es del 30% aproximadamente.

I.5 VALIDACIÓN E IMPUTACIÓN AUTOMÁTICAS.

A lo largo de los años las agencias estadísticas federales han buscado conciliar tres aspectos en el proceso de validación e imputación: reducir su costo, aumentar su calidad y acelerarlo con el fin de producir resultados más rápidamente. Por esta razón, muchas agencias están desarrollando técnicas computacionales para dicho efecto. Una de estas técnicas es la validación e imputación automáticas. Tradicionalmente, la corrección de la información

entre las etapas de entrada de datos y tabulación se ha realizado manualmente resultando costosa y lenta. Esta corrección es importante ya que la información faltante, inválida o inconsistente puede crear muchos problemas en la producción y análisis de tablas.

La validación e imputación automáticas ofrece ventajas metodológicas en comparación con la manual ya que esta última hace al proceso menos repetible y más sujeto a la crítica pues diferentes personas pueden hacer diferentes cosas en situaciones similares. Cuando el trabajo se hace en papel, es difícil dar un seguimiento y es imposible estimar el efecto de las acciones de validación. Finalmente, algunas tareas están por encima de la capacidad de los validadores humanos. Por ejemplo, puede que sea imposible para una persona el mantener una estructura de frecuencia multivariada de los datos cuando se hacen cambios.

La computadora garantiza la verificación y corrección uniforme de todos los registros de la encuesta de acuerdo con las instrucciones dadas por el sistema. El empleo de un personal numeroso involucra el manejo diferenciado de los registros y crea nuevas fuentes de error. También las computadoras han permitido detectar muchos de los errores que antes eran imposibles de detectar. Así, los registros individuales (y por ende los resultados) son corregidos más extensamente de lo que antes se podía.

Hasta ahora, muchos de los procedimientos automáticos desarrollos son hechos a la medida para encuestas especificas, por lo que tanto el desarrollo como la implementación son costosos. Mucho de este costo podría ser ahorrado si un sistema o programa de computadora pudiera tener varias aplicaciones.

El desarrollo de un sistema de validación e imputación generalizado deberá abarcar los siguientes cinco componentes de validación e imputación:

a) Especificación de las reglas de validación: dichas reglas son usadas para detectar errores en la información.

b) Análisis de las reglas: es la función de verificar que las reglas especificadas han sido definidas correctamente usando varios diagnósticos.

c) Aplicación de las reglas: es una función que evalúa cada registro de acuerdo a estas reglas.

d) Localización de error: identifica los elementos de los datos que tienen que ser imputados de modo que después de la imputación la información de los registros satisfaga todas las reglas.

e) Imputación: asigna valores para aquellos registros que necesitan corrección.

La especificación de las reglas y su análisis pueden ser desempeñados antes de la recolección de información.

Nuestro trabajo se enfoca en los incisos c) – e), ya que se aplican repetitivamente a lo largo del proceso de validación e imputación (al menos una vez para cada registro) y representan

la mayor parte, en tiempo y dinero, de dicho proceso, lo que hace deseable su automatización.

I.6 PROBLEMAS ACTUALES DE LA VALIDACIÓN Y LA IMPUTACIÓN AUTOMÁTICAS.

El uso de las computadoras ha producido un número considerable de nuevos problemas debido a que muchos de los sistemas hasta ahora desarrollados no pueden conciliar las siguientes dos características deseables en un sistema de validación:

i) Reconocer los patrones particulares de error así como las correlaciones de inconsistencia que surgen en una encuesta particular.

ii) Poseer una lógica comparativamente simple y fácilmente programable, pues de lo contrario habría una tendencia a no utilizar validación automática.

La característica i) requiere de un sistema 'hecho a la medida' para la encuesta particular y por lo tanto utiliza una lógica específicamente orientada a dicho estudio; esto requiere un esfuerzo considerable de planeación y programación para cada encuesta. La característica ii) requiere de un sistema que pueda ser fácilmente entendido y aplicado por el programador; pero entonces tenderá a pasar por alto un escrutinio detallado de los patrones de error específicos.

Además, tres problemas son comunes a muchos sistemas que imputan registros:

iii) La corrección puede cambiar la distribución de una variable.

iv) En muchos casos no es posible identificar qué variable debe ser cambiada.

v) Una corrección puede ser la causa de un nuevo error.

La no resolución de iii) afectará las estadísticas que se obtendrán de los datos en procesos subsecuentes de la encuesta, reduciendo así la confiabilidad de los resultados. En cuanto a iv), suele haber casos en los que diferentes combinaciones de variables, al ser cambiadas, llevan a un registro válido, pero de entre ellas es de desear identificar la que menos lo altera. La ocurrencia de v) requeriría la repetición del ciclo de validación e imputación, lo cual, además de implicar mayores costos, no garantiza la obtención de un registro válido.

Nuestro trabajo intenta resolver los problemas anteriores mediante una teoría que concilia las características i) e ii), mantiene las distribuciones marginales de las variables e identifica la imputación óptima que resulte en un registro válido.

I.7 OBJETIVO DE LA TESIS.

Debido a que actualmente en el (INEGI) existen únicamente sistemas de validación e imputación automáticos específicos a encuestas particulares, se requiere, en primera instancia, de trabajos teóricos que sienten las bases para la construcción de un sistema general de validación e imputación. En segunda instancia, se requiere de los algoritmos que permitan el paso de la teoría a la implementación computacional. Dichos algoritmos incorporan los resultados teóricos a una estructura que, por sus características particulares, son fáciles de programar. Es con este fin que presentamos el siguiente trabajo.

I.8 RESUMEN DE LA TESIS.

Un *edit* es una regla específica de validación. Se dice que un registro lo falla cuando sus valores cumplen esta regla. Para facilitar el uso de los *edits* se estandarizan a una forma normal. Los *edits* pueden ser de dos formas: explícitos (especificados por los especialistas en la encuesta y esenciales para la validación) e implícitos (generados a partir de los *edits* previos).

De entre los implícitos, los que son fundamentales para la imputación son los denominados esencialmente nuevos no redundantes porque, junto con los explícitos, nos permiten identificar los campos a imputar en un registro que falla *edits*.

A la unión de los *edits* explícitos y los implícitos esencialmente nuevos no redundantes se le llama el conjunto completo de *edits* (CCE). Debido a que estos últimos no están disponibles desde el inicio, deben ser generados. I. P. Fellegi y D. Holt (1976) dan las bases teóricas para generar dicho conjunto, pero no una manera sistemática de obtenerlo; por lo que se desarrollaron algoritmos con este propósito (algoritmos GKL1, WW y BCC).

El Algoritmo GKL1 permite, de manera ordenada, obtener el CCE. El Algoritmo WW, comparado con el anterior, reduce los cálculos necesarios para obtener dicho conjunto.

Los anteriores algoritmos no consideran el problema de *set covering* (PSC), el cual, aunque en sí mismo se puede utilizar para obtener el CCE, hace su uso poco eficiente debido a que el número de restricciones que incorpora va creciendo a medida que se van generando nuevos *edits* esencialmente nuevos no redundantes. El Algoritmo BCC logra incorporar, de manera eficiente, el PSC al mantener siempre reducido el número de restricciones.

Una vez obtenido, el CCE facilita la imputación de los registros. El Algoritmo FH nos da el conjunto mínimo de campos a imputar (CMCI) para cada registro. Una vez obtenido este conjunto, se aplican los Algoritmos de Imputación Secuencial y Conjunta. El primero encuentra, por cada campo del CMCI, un registro que pasó todos los *edits* e imputa el valor del campo en el registro erróneo. El de Imputación Conjunta encuentra un solo registro que pasó todos los *edits* e imputa los valores del CMCI en el registro erróneo.

Debido a que la obtención del CCE puede llegar a ser muy costosa, se desarrollaron algoritmos que logran imputar registros en ausencia del CCE. De contar únicamente con los *edits* explícitos, el Algoritmo GKL2 encuentra, para cada registro, los *edits* implícitos

suficientes para obtener la imputación óptima. En caso de que se cuente además con los *edits* implícitos de primer nivel (generados a partir de *edits* explícitos), los Algoritmos CW encuentran la imputación óptima más rápidamente que el Algoritmo GKL2. El Algoritmo CW2 es más rápido que el CW1 ya que no genera *edits* implícitos.

La metodología de imputación del vecino más cercano (MIVC) ofrece un enfoque diferente: en lugar de encontrar primero el CMCI y luego el(los) registro(s) a usar en la imputación (como en los Algoritmos de Imputación Secuencial y Conjunta), primero encuentra los registros a utilizarse y luego imputa los campos necesarios para obtener un registro válido.

Lo anterior se aplica a *edits* lógicos (que involucran datos cualitativos). Para los *edits* aritméticos (que involucran datos cuantitativos) de tipo lineal, se desarrolló teoría básica similar a la de los lógicos.

Con el objetivo de que los resultados sean fácilmente aplicados a nivel computacional, *edits*, registros y sus operaciones pueden expresarse con matrices binarias.

Todos los resultados cuentan con bases teóricas que los sustentan.

CAPÍTULO I

LOS *EDITS* COMO INSTRUMENTOS PARA LA VALIDACIÓN

A una regla específica de validación la llamaremos ***edit***, término que se utilizará a lo largo de este trabajo. Decimos que un registro ***falla*** un *edit* cuando los valores del registro cumplen la regla de validación y de hacerlo el registro será erróneo.

Cuando algún registro falla algún *edit*, teóricamente tenemos cinco opciones para solucionar el problema:

1. Revisar el cuestionario original, esperando que la información contenida en éste sea la correcta.

2. Recontactar al encuestado para obtener la respuesta correcta.

3. Hacer una imputación manual, realizada por un equipo de especialistas en la encuesta.

4. Hacer una imputación mediante un software especializado.

5. Eliminar los registros que fallen cualquier *edit* o al menos omitirlos de los análisis que empleen los campos no válidos, suponiendo que las inferencias estadísticas no se verán afectadas por estas omisiones.

En cuanto a la imputación de valores en campos de un registro, desearemos usar las partes válidas del registro e imputar los menos campos posibles que permitan al registro pasar todos los *edits*. Las correcciones del tipo 1 deben minimizarse mediante la aplicación rigurosa de controles de calidad a los procesos de entrada de los datos. La corrección del tipo 2 generalmente no puede realizarse ya que en la mayoría de los casos los encuestados no pueden ser recontactados debido a las limitaciones de tiempo y dinero. Entre las correcciones del tipo 3 y 4 es preferible la corrección del tipo 4, pues las computadoras aplican los *edits* en forma rápida y consistente. Sin embargo, la complejidad en la programación y la rigidez de los programas pueden ser desventajas. Este trabajo trata de resolver estos problemas:

- Con respecto a la complejidad, simplifica la tarea de programar, al identificar una forma simple y uniforme de presentar todos los *edits*.

- En cuanto a la rigidez, descompone los *edits* en series de reglas simples y no relacionadas con una forma común.

- Adicionalmente, permite deducir automáticamente de los *edits* las correcciones necesarias.

Nos centraremos en los datos cualitativos de una encuesta, aunque algunos resultados se pueden también aplicar a los datos cuantitativos.

Consideraremos tres criterios para la imputación de datos cualitativos:

1. Las imputaciones en cada registro deben hacerse para que los nuevos datos satisfagan todos los *edits* mediante el cambio de los menos elementos posibles.

2. Las imputaciones deben derivarse automáticamente de los *edits*.

3. Se deben mantener, tanto como sea posible, las distribuciones marginales (preferentemente la distribución conjunta) de las variables.

Diferenciamos *edits* de dos tipos:

- ***Lógicos***: son los que involucran datos cualitativos.

- ***Aritméticos***: son los que involucran datos cuantitativos y tendrán sentido sólo en presencia de una métrica.

1.1 LA FORMA NORMAL DE LOS *EDITS* LÓGICOS.

Supongamos que tenemos un registro con N campos.

Entenderemos por:

- A_i Conjunto de los valores aceptables en el campo i, con $i = 1, \ldots, N$.

Ejemplo 1.1:

Supongamos que tenemos un cuestionario en el que cada registro tiene 5 entradas.

$$\text{Sexo}: A_1 = \{\{\text{Masculino}\}, \{\text{Femenino}\}\}$$
$$\text{Edad}: A_2 = \{\{0-14\}, \{15-16\}, \{17+\}\}$$
$$\text{Educación}: A_3 = \{\{\text{Ninguna}\}, \{\text{Primaria}\}, \{\text{Secundaria}\}, \{\text{Media Superior}\}\}$$
$$\text{Relación con el jefe de familia}: A_4 = \{\{\text{Jefe}\}, \{\text{Cónyuge}\}, \{\text{Hijo o Hija}\}, \{\text{Otro}\}\}$$
$$\text{Estado civil}: A_5 = \{\{\text{Soltero}\}, \{\text{Casado}\}, \{\text{Divorciado}\}, \{\text{Separado}\}, \{\text{Viudo}\}\}$$

- n_i Número de elementos en el conjunto A_i, con $i = 1, \ldots, N$.

Ejemplo 1.2: $\text{Edad}: n_2 = 3$.

- $A = \prod_{i=1}^{N} A_i$ Conjunto de todos los valores aceptables para un registro.

Ejemplo 1.3:

$$A = A_1 \times A_2 \times A_3 \times A_4 \times A_5.$$

Para simplificar la notación, en los ejemplos subsecuentes, utilizaremos los símbolos $\langle\,\rangle$ para denotar el producto cartesiano. Así,

$$A = \langle A_1, A_2, A_3, A_4, A_5 \rangle.$$

- A_i^r Subconjunto de A_i que contiene los valores no válidos del campo i en el *edit* r, con $i = 1, \ldots, N$.

Ejemplo 1.4:

Si sabemos que la población que tiene entre 0 y 14 años no puede ser jefe de familia, entonces podemos formar un *edit* que denotaremos con el subíndice 1, para el que

$$A_1^1 = A_1, A_2^1 = \{0-14\}, A_3^1 = A_3, A_4^1 = \{\text{Jefe}\}, A_5^1 = A_5.$$

- $y_i \in A_i^r$ El registro y tiene en el campo i un valor perteneciente a A_i^r.

Ejemplo 1.5:

Supongamos que el registro y tiene las siguientes entradas:

Sexo:	Masculino
Edad:	14
Educación:	Secundaria
Relación con el Jefe de Familia:	Jefe
Estado Civil:	Soltero

Entonces $y_1 \in A_1^1$, $y_2 \in A_2^1$, $y_3 \in A_3^1$, $y_4 \in A_4^1$ y $y_5 \in A_5^1$.

- $E_r = f(A_1^r, A_2^r, ..., A_N^r)$ Conjunto de los valores que no son aceptables en el *edit r* especificado por *f*, donde *f* es una función que conecta a estos conjuntos mediante el producto cartesiano.

Ejemplo 1.6: $E_1 = f(A_1^1, A_2^1, A_3^1, A_4^1, A_5^1) = \prod_{i=1}^{5} A_i^1 = \langle A_1^1, A_2^1, A_3^1, A_4^1, A_5^1 \rangle$.

- $y \in E_r$ El registro y pertenece al conjunto de los valores no aceptables, es decir, el registro y falla el *edit r*.

Ejemplo 1.7:

Si un registro y especifica que el individuo tiene 14 años y es jefe de familia, entonces:

$$y \in \langle A_2^1, A_4^1 \rangle .$$

Por lo que ahora es fácil mostrar que:

$$y \in f \Leftrightarrow y \in A_{i_1}^r \times \cdots \times A_{i_{m_1}}^r$$

para algún *edit r* formado por un subconjunto finito de campos $\{i_1, ..., i_{m_1}\}$ involucrados en este *edit*.

Es decir, $y \in f$ si y sólo si y pertenece al producto cartesiano de los conjuntos A_1^r, $A_2^r, ..., A_N^r$. Entonces la expresión anterior es equivalente a:

$$y \in f \Leftrightarrow y \in \prod_{i \in S} A_i^r \qquad \text{donde } S \text{ es el conjunto de índices } \{i_1, ..., i_{m_1}\}.$$

Ya que cualquier *edit* puede ser descompuesto en una serie de enunciados de la forma: "una combinación especifica de valores no es permitida", entonces un *edit* e_r se expresa como

$$e_r : \prod_{i=1}^{N} A_i^r = F$$

donde $A_i^r \neq \varnothing \; \forall i$, $A_i^r = A_i \; \forall i \notin S$ y F representa la falla del registro.

Establecemos lo anterior para incluir a todos los campos del registro y hacemos $A_i^r = A_i$ para que el campo i que no está en S no se vea involucrado en el *edit*, únicamente los campos que pertenecen al conjunto S (es decir, A_i^r será un subconjunto propio no vacío de A_i para toda $i \in S$).

Entonces e_r puede ser expresado como

$$e_r : E_r = F \qquad \text{donde } E_r = \prod_{i=1}^{N} A_i^r .$$

A esta forma de representar un *edit* se le llamará la ***forma normal del edit***.

Ejemplo 1.8:

Supongamos que un especialista en la encuesta especifica el siguiente *edit*: "Si la edad de una persona es de 14 años o menos o es un estudiante de primaria, entonces la relación con el jefe de familia no debe de ser de jefe y el estado civil debe ser soltero." Observamos que en este *edit* el sexo no está involucrado.

Mediante los siguientes pasos este *edit* puede convertirse a su forma normal:

1. Si [(Edad = 0-14) ó (Educación = Primaria)] entonces debe pasar [(Relación con el jefe de familia ≠ Jefe) y (Estado civil = Soltero)].

2. Si [(Edad = 0-14) ó (Educación = Primaria)] y no pasa que [(Relación con el jefe de familia ≠ Jefe) y (Estado civil = Soltero)] entonces el registro falla el *edit*.

3. Si $\left[\langle A_1, \{0-14\}, A_3, A_4, A_5\rangle \;\text{ ó }\; \langle A_1, A_2, \{\text{Primaria}\}, A_4, A_5\rangle\right]$ y no pasa que $\left[\langle A_1, A_2, A_3, \{\neq \text{Jefe}\}, A_5\rangle \;\text{ y }\; \langle A_1, A_2, A_3, A_4, \{\text{Soltero}\}\rangle\right]$ entonces el registro falla el edit.

4. Si $\left[\langle A_1, \{0-14\}, A_3, A_4, A_5\rangle \;\text{ ó }\; \langle A_1, A_2, \{\text{Primaria}\}, A_4, A_5\rangle\right]$ y $\left[\text{no pasa } \langle A_1, A_2, A_3, \{\neq \text{Jefe}\}, A_5\rangle \text{ o no pasa } \langle A_1, A_2, A_3, A_4, \{\text{Soltero}\}\rangle\right]$ entonces el registro falla el edit.

5. $$e:\left[\langle A_1,\{0-14\},A_3,A_4,A_5\rangle \cup \langle A_1,A_2,\{\text{Primaria}\},A_4,A_5\rangle\right]\cap \left[\langle A_1,A_2,A_3,\{\text{Jefe}\},A_5\rangle \cup \langle A_1,A_2,A_3,A_4,\{\neq \text{Soltero}\}\rangle\right]=F$$

6. $$e:\left\{\langle A_1,\{0-14\},A_3,A_4,A_5\rangle \cap\left[\langle A_1,A_2,A_3,\{\text{Jefe}\},A_5\rangle \cup \langle A_1,A_2,A_3,A_4,\{\neq \text{Soltero}\}\rangle\right]\right\}\cup \left\{\langle A_1,A_2,\{\text{Primaria}\},A_4,A_5\rangle \cap\left[\langle A_1,A_2,A_3,\{\text{Jefe}\},A_5\rangle \cup \langle A_1,A_2,A_3,A_4,\{\neq \text{Soltero}\}\rangle\right]\right\}=F$$

7. $$\begin{aligned} e:&\left[\langle A_1,\{0-14\},A_3,A_4,A_5\rangle \cap \langle A_1,A_2,A_3,\{\text{Jefe}\},A_5\rangle\right]\cup \\ &\left[\langle A_1,\{0-14\},A_3,A_4,A_5\rangle \cap \langle A_1,A_2,A_3,A_4,\{\neq \text{Soltero}\}\rangle\right]\cup \\ &\left[\langle A_1,A_2,\{\text{Primaria}\},A_4,A_5\rangle \cap \langle A_1,A_2,A_3,\{\text{Jefe}\},A_5\rangle\right]\cup \\ &\left[\langle A_1,A_2,\{\text{Primaria}\},A_4,A_5\rangle \cap \langle A_1,A_2,A_3,A_4,\{\neq \text{Soltero}\}\rangle\right]=F \end{aligned}$$

8. $$\begin{aligned} e_1&:\langle A_1,\{0-14\},A_3,A_4,A_5\rangle \cap \langle A_1,A_2,A_3,\{\text{Jefe}\},A_5\rangle=F \\ e_2&:\langle A_1,\{0-14\},A_3,A_4,A_5\rangle \cap \langle A_1,A_2,A_3,A_4,\{\neq \text{Soltero}\}\rangle=F \\ e_3&:\langle A_1,A_2,\{\text{Primaria}\},A_4,A_5\rangle \cap \langle A_1,A_2,A_3,\{\text{Jefe}\},A_5\rangle=F \\ e_4&:\langle A_1,A_2,\{\text{Primaria}\},A_4,A_5\rangle \cap \langle A_1,A_2,A_3,A_4,\{\neq \text{Soltero}\}\rangle=F \end{aligned}$$

9. $$\begin{aligned} e_1&:\langle A_1,\{0-14\},A_3,\{\text{Jefe}\},A_5\rangle=F \\ e_2&:\langle A_1,\{0-14\},A_3,A_4,\{\neq \text{Soltero}\}\rangle=F \\ e_3&:\langle A_1,A_2,\{\text{Primaria}\},\{\text{Jefe}\},A_5\rangle=F \\ e_4&:\langle A_1,A_2,\{\text{Primaria}\},A_4,\{\neq \text{Soltero}\}\rangle=F \end{aligned}$$

Estos cuatro *edits* juntos son equivalentes al *edit* original y están escritos en la forma normal.

Entenderemos por ***edits explícitos*** al conjunto de *edits* especificados por los especialistas en la materia.

Sea $\underline{e}_E=\{e_1,\ldots,e_m\}$ el conjunto de *edits* explícitos y definamos

$$\underline{E}_E=\bigcup_{e_k\in \underline{e}_E} E_k.$$

Supondremos que todo *edit* e_k estará dado en su forma normal:

$$e_k:\prod_{j=1}^{n} A_j^k=F \quad \text{con} \quad A_j^k\neq\varnothing \text{ y } A_j^k\subseteq A_j\ \forall j.$$

Además, el campo *j* se dirá que ***entra*** en e_k si $A_j^k \neq A_j$.

Los *edits* explícitos suelen presentarse de dos formas:

- ***Edit simple***: establece que los valores permitidos para el campo *i* están dados por el conjunto A_i, cualquier otro valor es erróneo.

  Para convertir este *edit* a la forma normal basta agregar al conjunto A_i el elemento $\{c_i\}$, el cual será asignado durante la codificación a cualquier valor erróneo (es decir, será una bandera de que el dato en ese campo es erróneo). Así, el *edit* en su forma normal para cada campo será el siguiente:

  $$e: A_1 \times \cdots \times A_{i-1} \times \{c_i\} \times A_{i+1} \times \cdots \times A_n = F, \qquad i = 1, 2, ..., N.$$

  Ejemplo 1.9:

  'Edad': $A_2 = \{\{-1\}, \{0-14\}, \{15-16\}, \{17+\}\}$ donde $\{-1\}$ denota un valor erróneo.

  Si tenemos un registro con algún valor distinto, le asignamos el valor $\{-1\}$, que servirá de bandera para indicar que el valor en ese registro es erróneo.

  Entonces el *edit* en su forma normal será:

  $$e: \langle A_1, \{-1\}, A_3, A_4, A_5 \rangle = F$$

- ***Edit de consistencia:*** involucra a un conjunto finito de valores de la forma $f(A_1', A_2', ..., A_N')$ donde A_i' es un subconjunto de A_i; establece el siguiente enunciado:

  " Si $y \in f(A_1', A_2', ..., A_N')$ deberá suceder que $y \in g(A_1'', A_2'', ..., A_N'')$."

  Es decir, cuando un registro tiene ciertas combinaciones de valores en algunos campos, debe tener otras combinaciones de valores en otros campos.

  Para convertir este *edit* a la forma normal observemos que la expresión anterior es equivalente a:

  " Si $y \in f(A_1', A_2', ..., A_N')$ y $y \notin g(A_1'', A_2'', ..., A_N'')$ entonces hay un error. "

  Recordemos que *f* y *g* son productos cartesianos de conjuntos. Por lo tanto, tenemos que el enunciado se puede expresar como:

" Si $y \in f(A_1', A_2', ..., A_N')$ y $y \in g^c(A_1'', A_2'', ..., A_N'')$ entonces hay un error. "

donde $g^c(A_1'', A_2'', ..., A_N'')$ denota los complementos de A_i'' siempre y cuando A_i'' esté contenido propiamente en A_i.

Finalmente el *edit* nos queda de la siguiente forma:

$$e: f(A_1', A_2', ..., A_N') \cap g^c(A_1'', A_2'', ..., A_N'') = F.$$

Ejemplo 1.10:

Consideremos los siguientes subconjuntos:

$$A_2' = \{0-14\}, A_5' = \{\text{Soltero}\}.$$

Tenemos entonces el siguiente *edit* de consistencia:

" Si $y \in f(A_1, A_2', A_3, A_4, A_5)$ entonces $y \in g(A_1, A_2, A_3, A_4, A_5')$ "

donde f y g son los productos cartesianos de los conjuntos, es decir, el *edit* es:

"Si $y \in \langle A_1, A_2', A_3, A_4, A_5 \rangle$ entonces $y \in \langle A_1, A_2, A_3, A_4, A_5' \rangle$."

Por lo descrito anteriormente, el *edit* en su forma normal es:

$$\begin{aligned} e: & f(A_1, A_2', A_3, A_4, A_5) \cap g^c(A_1, A_2, A_3, A_4, A_5') \\ & = \langle A_1, A_2', A_3, A_4, A_5 \rangle \cap \langle A_1, A_2, A_3, A_4, A_5^{\complement} \rangle \\ & = \langle A_1 \cap A_1, A_2' \cap A_2, A_3 \cap A_3, A_4 \cap A_4, A_5 \cap A_5^{\complement} \rangle \\ & = \langle A_1, A_2', A_3, A_4, A_5^{\complement} \rangle = F. \end{aligned}$$

1.2 LA FORMA NORMAL DE LOS *EDITS* ARITMÉTICOS.

Aun cuando la forma normal no está concebida para los *edits* aritméticos, éstos se pueden expresar de esa forma.

Presentaremos ahora un procedimiento para los *edits* aritméticos que involucran únicamente expresiones lineales, es decir, de la forma:

$0 \geq M(a_1, a_2, ..., a_N)$ donde a_i es la variable correspondiente al campo i que toma valores en A_i (i = 1,..., N) y donde M es una función lineal.

La desigualdad puede ser cualquier otra ($>$, $\leq$ ó $<$).

Supongamos que $a_1 > 0$, entonces podemos despejar a_1 quedándonos la expresión:

$$a_1 \leq L(a_2, a_3, ..., a_N) \qquad \text{donde } L \text{ es también una función lineal.}$$

Entonces esta desigualdad se puede expresar como un *edit* en la forma normal:

$$A_1^0 \times A_2^0 \times \cdots \times A_N^0 = F \qquad \text{donde } A_1^0 = \{a_1 \mid a_1 \leq L(a_2, a_3, ..., a_N)\} \text{ y}$$
$$A_i^0 = \{a_i\},\ i = 2, ..., N.$$

Es decir, A_1^0 contiene a todos los valores de la variable a_1 que cumplen la desigualdad y A_i^0 contiene todos los valores posibles de la variable a_i ($i = 2, \ldots, N$).

Por otra parte, cuando en una encuesta tenemos valores cuantitativos podemos dividir en un número finito de rangos el espacio de valores posibles del campo. Así los datos cuantitativos podrán ser tratados como cualitativos para propósitos de validación e imputación.

Finalmente, por convención, si un registro contiene una entrada no válida en un campo, consideraremos que el registro falla todos los *edits* que involucren dicho campo.

1.3 EL CONJUNTO COMPLETO DE *EDITS*.

En la práctica generalmente se conocen sólo los *edits* fallados pero no los campos que causan estas fallas y cuyo cambio podría eliminarlas. Si queremos identificar el campo a cambiar debemos de considerar los *edits* implícitos que se construyen a partir de los *edits* explícitos. Entenderemos por ***edits implícitos*** al conjunto de *edits* cuya falla implique la falla de alguno de los *edits* explícitos.

Ejemplo 1.11:

Supongamos que hay dos *edits*:

$$e_1 : \langle A_1, \{0-14\}, A_3, A_4, \text{Alguna vez casado} \rangle = F$$
$$e_2 : \langle A_1, A_2, A_3, \{\text{Cónyuge}\}, \text{No está casado actualmente} \rangle = F$$

Debemos notar que *Alguna vez casado* representa al subconjunto del Estado civil {{Casado}, {Divorciado}, {Viudo}, {Separado}}, y *No estar casado actualmente* representa al subconjunto {{Soltero}, {Divorciado}, {Viudo}, {Separado}}.

Ahora supongamos que un registro tiene los siguientes valores:

Sexo :	Masculino
Edad :	14
Educación :	Secundaria
Relación con el jefe de familia :	Cónyuge
Estado civil:	Casado

Este registro falla el *edit* e_1 y pasa el *edit* e_2, por lo que si queremos corregir este registro mediante imputación debemos considerar cambiar el campo Estado civil. Es fácil verificar que dejando Edad y Relación con el jefe de familia sin cambio, todos los posibles valores del Estado civil resultarían en un registro que fallaría uno u otro de los dos *edits*. De hecho en este sencillo ejemplo es intuitivamente claro que hay un conflicto entre la Edad = 14 años y la Relación con el jefe de familia = Cónyuge. La existencia de este conflicto es implicada en forma lógica de los dos *edits*.

Los *edits* e_1 y e_2 pueden expresarse usando la flecha $\Rightarrow$ (que significa *implica*) como:

e_1 : (Edad = 0-14) $\Rightarrow$ (Estado civil = Soltero)

e_2 : (Estado civil = No está casado actualmente) $\Rightarrow$ (Relación con el jefe de familia = No cónyuge)

Entonces es obvio que:

(Estado civil = Soltero) $\Rightarrow$ (Estado civil = No está casado actualmente).

Y combinando e_1 y e_2 obtenemos:

(Edad = 0-14) $\Rightarrow$ (Relación con el jefe de familia = No cónyuge).

Finalmente podemos concluir:

$$e_3 : \langle A_1, \{0-14\}, A_3, \{\text{Cónyuge}\}, A_5 \rangle = F$$

Así los *edits* e_1 y e_2 implican lógicamente al *edit* e_3.

Provisionalmente, entenderemos por el conjunto completo de *edits* al conjunto de *edits* con la propiedad de que no puede derivarse de ellos ningún *edit* nuevo y lo denotaremos como $\underline{e}_C$. Una definición más formal se dará en el capítulo siguiente.

El punto a notar en este ejemplo es la importancia de identificar, junto con los *edits* iniciales e_1 y e_2, el *edit* implícito e_3. Los *edits* e_1 y e_2 sirven únicamente para identificar si el registro actual tiene problemas. Sin embargo, después de identificar el *edit* implícito e_3, podremos determinar sistemáticamente el (los) campo(s) que deberemos de cambiar para corregir las inconsistencias.

Ejemplo 1.12:

Supongamos que un cuestionario tiene cuatro campos cuantitativos. La información registrada en estos campos está denotada por las variables a, b, c y d respectivamente.

Supongamos que hay dos *edits* aritméticos conectando a estas variables, cada uno indicando una condición que debe cumplir el registro.

$e_1 : a + c + d \leq b,$ que es equivalente a, $a - b + c + d \leq 0$

$e_2 : 2b \leq a + 3c,$ que es equivalente a, $-a + 2b - 3c \leq 0$

Supongamos que el registro actual contiene los valores: $a = 3$, $b = 4$, $c = 6$, $d = 1$. Es fácil ver que el primer *edit* no pasa y el segundo sí. No es claro, aún en este sencillo ejemplo, qué campos (variables) necesitamos cambiar para satisfacer los *edits*. Sin embargo, si escribimos los *edits* implícitos, la situación se puede clarificar inmediatamente. Es fácil verificar que las siguientes tres desigualdades son implicadas por las dos anteriores:

$e_3 : b - 2c + d \leq 0,$ que se obtiene de sumar e_1 y e_2.

$e_4 : a - c + 2d \leq 0,$ que se obtiene de multiplicar e_1 por 2 y sumar e_2 al resultado.

$e_5 : 2a - b + 3d \leq 0,$ que se obtiene de multiplicar e_1 por 3 y sumar e_2 al resultado.

Fácilmente podemos verificar que los *edits* e_1, e_3 y e_4 fallan y e_2 y e_5 pasan. Observemos que el campo c está involucrado en los tres *edits* que fallan, por lo que bastaría imputar este campo para que el registro pase todos los *edits*.

En general, buscaremos identificar un conjunto mínimo de campos que, al ser modificados hagan que se satisfagan todos los *edits*.

1.4 CONVERSIÓN A TEORÍA MATRICIAL.

Presentaremos en esta sección métodos prácticos para la conversión de las secciones anteriores a la teoría matricial.

La clave para la implementación de un sistema automático de validación e imputación consiste en la posibilidad de representar los *edits* lógicos en forma de líneas de 0's y 1's. El método para lograr esto se presentará usando el siguiente ejemplo.

Ejemplo 1.13:

Supongamos que un registro contiene cinco campos, cada uno, con estos posibles conjuntos de valores:

Sexo	Edad	Educación	Relación con el jefe de familia	Estado civil
Masculino	0-14	Ninguna	Esposa	Soltero
Femenino	15-16	Primaria	Esposo	Casado
	17+	Secundaria	Hijo o hija	Divorciado
		Media superior	Otro	Separado
				Viudo

Observamos que el único campo que cambia con relación a los ejemplos anteriores es el de Relación con el jefe de familia que en lugar de contener a Jefe y Cónyuge tiene ahora Esposa y Esposo.

Supongamos también que tenemos un conjunto completo de *edits*:

$$e_1 : \langle \{\text{Masculino}\}, A_2, A_3, \{\text{Esposa}\}, A_5 \rangle = F$$

$$e_2 : \langle A_1, \{0-14\}, A_3, A_4, \text{Alguna vez casado} \rangle = F$$

$$e_3 : \langle A_1, A_2, A_3, \{\{\text{Esposa}\}, \{\text{Esposo}\}\}, \text{No está casado actualmente} \rangle = F$$

$$e_4 : \langle A_1, \{0-14\}, A_3, \{\{\text{Esposa}\}, \{\text{Esposo}\}\}, A_5 \rangle = F$$

$$e_5 : \langle A_1, \{\{0-14\}, \{15-16\}\}, \{\text{Media superior}\}, A_4, A_5 \rangle = F$$

Se da un registro que contiene las siguientes características:

Sexo :	Masculino
Edad :	12
Educación :	Primaria
Relación con el jefe de familia :	Esposa
Estado civil :	Casado

Consideremos la Tabla 1.1. El encabezado de la tabla corresponde a la descripción de todos los posibles valores de todos los posibles campos: una columna corresponde a cada posible valor, un conjunto de columnas corresponde a un campo.

Edit	**Sexo**		**Edad**			**Educación**				**Relación con el jefe de familia**				**Estado civil**				
	Masculino	Femenino	0-14	15-16	17+	Ninguna	Primaria	Secundaria	Media superior	Esposa	Esposo	Hijo (a)	Otro	Soltero	Casado	Divorciado	Separado	Viudo
e_1	1	0	1	1	1	1	1	1	1	1	0	0	0	1	1	1	1	1
e_2	1	1	1	0	0	1	1	1	1	1	1	1	1	0	1	1	1	1
e_3	1	1	1	1	1	1	1	1	1	1	1	0	0	1	0	1	1	1
e_4	1	1	1	0	0	1	1	1	1	1	1	0	0	1	1	1	1	1
e_5	1	1	1	1	0	0	0	0	1	1	1	1	1	1	1	1	1	1
Registro actual	1	0	1	0	0	0	1	0	0	1	0	0	0	0	1	0	0	0

Tabla 1.1. *Matriz para la conversión del Ejemplo 1.13.*

Cada *edit* en la forma normal puede ser completamente caracterizado en términos del conjunto de valores en cada campo que se ven involucrados en ese *edit*. Por ejemplo, tomemos e_3 :

$$e_3 : \langle A_1, A_2, A_3, \{\{\text{Esposa}\}, \{\text{Esposo}\}\}, \text{No está casado actualmente}\rangle = F$$

Este *edit* está totalmente caracterizado por los 5 conjuntos de valores que corresponden a los 5 campos. Estos conjuntos de valores, están caracterizados en la Tabla 1.1 por una entrada de 1 para cada miembro del conjunto y 0 en otro caso. Entonces, para e_3, ponemos 1's en las columnas correspondientes a los dos posibles valores de Sexo, los tres posibles valores de Edad, los cuatro posibles valores de Educación; sin embargo, de las cuatro columnas correspondientes a los valores de Relación con el jefe de familia colocamos 1 sólo en las dos columnas de Esposo y Esposa; poniendo 0's en las otras dos columnas y de los 5 posibles valores de Estado civil, sólo las cuatro columnas correspondientes a la descripción No casado actualmente son completadas con 1; en la columna Casado colocamos un 0.

La Tabla 1.1 provee así una representación única de los cinco *edits*.

A la matriz de 0's y 1's ejemplificada por la Tabla 1.1 se le llama la ***matriz de edits lógicos***.

Similarmente podemos convertir un registro en una línea de 0's y 1's. Basta asignar en un campo un 1 al valor que posee el registro y 0's a los demás valores del campo (así deberá haber un 1 por cada campo). La codificación del registro de nuestro ejemplo se presenta en la última fila de la Tabla 1.1.

La aplicación del *edit* al registro se puede hacer de dos formas:

A. Si el producto del *edit* y el registro es igual al número de campos entonces el *edit* falla.

Ejemplo 1.14:

Tomando e_1 para validar tenemos:

e_1	1	0	1	1	1	1	1	1	1	1	0	0	0	1	1	1	1	1
Registro	1	0	1	0	0	0	1	0	0	1	0	0	0	0	1	0	0	0
e_1 × Registro	1	0	1	0	0	0	1	0	0	1	0	0	0	0	1	0	0	0

Entonces definimos el producto como la suma de los elementos de la última fila, que en este caso es igual a 5.

Como el producto coincide con el número de campos, nuestro registro falla e_1.

B. Si tomamos el valor del *edit* en las columnas donde el registro tiene 1 y hacemos el producto de los valores seleccionados, entonces el *edit* falla si el producto es 1.

Ejemplo 1.15:

Considerando e_2 y e_3 para validar, tomamos de la matriz las columnas en las que el registro tiene 1 y agregamos una columna que corresponde al producto de los elementos en la fila:

Edit	**Sexo**	**Edad**	**Educación**	**Relación con el Jefe de Familia**	**Estado Civil**	**Producto**
	Masculino	0-14	Primaria	Esposa	Casado	
e_2	1	1	1	1	1	1
e_3	1	1	1	1	0	0

Como el producto para e_2 es 1 y el de e_3 es 0, el registro falla e_2 y pasa e_3.

Observamos que si un *edit* tiene 1's en todos los valores de un campo, este campo no intervendrá en él. Además, ningún *edit* puede tener 0's en todos los valores de un campo, pues esto implicaría que ningún registro podría fallarlo.

Finalmente, si se desean incluir *edits* simples basta agregar a cada campo una nueva columna cuyo valor será un valor de validez (e.g., -1). Si un *edit* es de este tipo en la Tabla 1.1 se pondrá 1 en el valor de validez, 0's en los demás valores de este campo y 1's en todos los valores de los otros campos.

Tenemos así una forma de representar con matrices binarias los *edits* y los registros de una encuesta. Además, dichas matrices nos permiten realizan de manera sencilla tanto las operaciones entre *edits* como las que se realizan entre un *edit* y un registro. Esta representación es un paso fundamental para el desarrollo de sistemas generales de validación e imputación automáticas.

CAPÍTULO II
GENERACIÓN DEL CONJUNTO COMPLETO DE *EDITS*

I. P. Fellegi y D. Holt (1976) presentan una metodología para validar e imputar datos. En dicho trabajo se desarrolla un cuerpo teórico que permite la obtención del conjunto completo de *edits* a partir de los *edits* explícitos. Recordemos que este conjunto es esencial para la validación e imputación de los registros de una encuesta.

En este capítulo se presentan resultados teóricos necesarios para la obtención del conjunto completo de *edits*, así como un método para su conversión a teoría matricial.

2.1 BASES TEÓRICAS EN LA OBTENCIÓN DEL CONJUNTO COMPLETO DE *EDITS* LÓGICOS.

En primer lugar, presentamos un lema que nos permitirá obtener *edits* implícitos de un conjunto de *edits* explícitos:

Lema 2.1:

Sea S un subconjunto de índices de *edits*. Si los *edits* $\{e_r\}_{r\in S}$ tienen forma normal:

$$e_r : \prod_{j=1}^{N} A_j^r = F, \ \forall r \in S$$

Entonces, tomando $i \in \{1, \ldots, N\}$,

$$e^* : \prod_{j=1}^{N} A_j^* = F \qquad \text{donde} \quad \begin{array}{l} A_j^* = \bigcap_{r \in S} A_j^r \text{ si } j \neq i \\ A_j^* = \bigcup_{r \in S} A_j^r \text{ si } j = i \end{array}$$

es un *edit* implícito válido siempre y cuando $A_j^* \neq \varnothing$, $\forall j$.

Demostración

Como $A_j^* \neq \varnothing \quad \forall j \in \{1,...,N\}$, e^* es un *edit* válido en su forma normal.

Para ver que es implícito supongamos que un registro $y = (y_1,...,y_N)$ falla e^*. Entonces $y_j \in A_j^* \quad \forall j \in \{1,...,N\}$.

Y por definición $y_j \in A_j^r \quad \forall r \in S \quad \forall j \neq i$ y $y_i \in A_i^s$ para alguna $s \in S$.

De aquí que $y_j \in A_j^s \quad \forall j \in \{1,...,N\}$ y así $y \in \prod_{j=1}^{N} A_j^s$ por lo tanto y falla e_s.

Concluimos que e^* es un *edit* implícito ya que su falla implica la falla de algún *edit* en S.■

A i se le denominará el ***campo generador*** y a $\{e_r\}_{r \in S}$ los ***edits contribuyentes***. De esta forma a e^* se le denotará $e^*\left(i, \{e_r\}_{r \in S}\right)$.

Ejemplo 2.1:

Retomemos el Ejemplo 1.11. Los *edits* e_1 y e_2 son:

$e_1 : \langle A_1, \{0-14\}, A_3, A_4, \text{Alguna vez casado} \rangle = F$

$e_2 : \langle A_1, A_2, A_3, \{\text{Cónyuge}\}, \text{No está casado actualmente} \rangle = F$

Sea el Estado civil (i = 5) el campo generador. Entonces, por el lema anterior:

A_5^* = Alguna vez casado $\bigcup$ No está casado actualmente = A_5.

Y para los otros cuatro campos:

$$A_1^* = A_1 \cap A_1 = A_1,$$
$$A_2^* = \{0-14\} \cap A_2 = \{0-14\},$$
$$A_3^* = A_3 \cap A_3 = A_3,$$
$$A_4^* = A_4 \cap \{\text{Cónyuge}\} = \{\text{Cónyuge}\}.$$

Entonces el *edit* implícito queda:

$$e^* : \langle A_1, \{0-14\}, A_3, \{\text{Cónyuge}\}, A_5 \rangle = F$$

Diremos que un *edit* implícito es ***esencialmente nuevo*** si $A_i^* = A_i$ (i, el campo generador) y $A_i^r \subset A_i \ \ \forall r \in S$ en el Lema 2.1. Éste es el caso del *edit* generado en el ejemplo anterior. Definimos que el *edit* e_r ***domina*** al *edit* e_q si $E_q \subseteq E_r$. Así, diremos que un *edit* es ***no redundante*** si ningún otro *edit* lo domina.

Al conjunto de los *edits* explícitos junto con todos los *edits* esencialmente nuevos no redundantes se le llamará el ***conjunto completo de edits*** y se denotará por $\underline{e}_C$. Para la generación de este conjunto será necesario tomar, en el Lema 2.1, todos los posibles subconjuntos tanto de *edits* explícitos como implícitos que se vayan generando como *edits* contribuyentes y todos los campos como campo generador.

Definimos $\underline{e}_K$ al subconjunto de $\underline{e}_C$ que involucra únicamente a los campos $1,...,K$, es decir, $\underline{e}_K$ consiste de los *edits*

$$e_r : \prod_{j=1}^{N} A_j^r = F \ \text{ tales que } A_j^r = A_j, \ j = K+1, K+2, ..., N.$$

Ejemplo 2.2:

En términos del Ejemplo 2.1, si e_1, e_2 y e^* forman un conjunto completo de *edits* $\underline{e}_C$, correspondiente a e_1 y e_2, entonces el subconjunto $\underline{e}_4$, que involucra a los campos 1, 2, 3 y 4, consiste únicamente de e^* (ya que e^* es el único *edit* que no involucra al campo 5). $\underline{e}_3$, el subconjunto de *edits* que involucran únicamente a los campos 1, 2 y 3, es nulo.

Presentamos ahora un teorema y dos corolarios que después nos serán útiles para la imputación de valores en los campos de un registro.

Teorema 2.1:

Suponga que se tiene un cuestionario con N preguntas, igual al número de variables. Entonces los valores reportados en el cuestionario por el individuo p serán denotados por el vector $y^p = (y_1^p, y_2^p, ..., y_N^p)$. Si y^p es un registro que

satisface todos los *edits* en $\underline{e}_{K-1}$, pero no así algún *edit* en $\underline{e}_K$, entonces existe un valor y_K^0 tal que $y^{p\prime} = (y_1^p, y_2^p, ..., y_{K-1}^p, y_K^0, y_{K+1}^p, ..., y_N^p)$ satisfará todos los *edits* en $\underline{e}_K$.

Demostración

Supongamos que el teorema es falso, entonces existe un registro $y^0 = (y_1^0, y_2^0, ..., y_N^0)$ que satisface todos los *edits* en $\underline{e}_{K-1}$, no satisface algún *edit* en $\underline{e}_K$ y ningún posible valor $y_K \in A_K$ hace que $y = (y_1^0, ..., y_{K-1}^0, y_K, y_{K+1}^0, ..., y_N^0)$ satisfaga todos los *edits* en $\underline{e}_K$. Entonces para todo $y_K \in A_K$, $y = (y_1^0, ..., y_{K-1}^0, y_K, y_{K+1}^0, ..., y_N^0)$ falla al menos un *edit* en $\underline{e}_K$.

Para cada valor posible $y_K \in A_K$ identifiquemos un *edit* fallado en $\underline{e}_K$ de la forma:

$$e_r : \prod_{i=1}^{N} A_i^r = F, \quad r = 1, 2, ..., R, \tag{1}$$

donde:

$y_i^0 \in A_i^r$, $i \in \{1, ..., K-1, K+1, ..., N\}$, $y_K \in A_K^r$ y $R \in \{1, ..., \min\{n_K, \#(\underline{e}_K)\}\}$.
Observemos que A_K^r no puede ser igual a A_K dado que, si así fuera, existiría un *edit* en $\underline{e}_{K-1}$ que sería fallado por $y^0 = (y_1^0, y_2^0, ..., y_N^0)$ contradiciendo así la hipótesis original.

Consideremos el siguiente *edit* implícito construido utilizando el Lema 2.1 con *edits* contribuyentes (1) y campo generador K,

$$e^* : \prod_{i=1}^{K-1} \left\{ \bigcap_{r=1}^{R} A_i^r \right\} \times \left\{ \bigcup_{r=1}^{R} A_K^r \right\} \times \prod_{i=K+1}^{N} A_i = F.$$

Como $y_i^0 \in A_i^r$, $i \in \{1, ..., K-1\}$ $\forall r \in \{1, ...R\}$, $\bigcap_{r=1}^{R} A_i^r \neq \varnothing$ $\forall i \in \{1, ..., K-1\}$.
Así e^* es un *edit* implícito válido. Además, de acuerdo a nuestra hipótesis, cada posible valor y_K del campo K está contenido en al menos uno de los conjuntos A_K^r, por lo que $\bigcup_{r=1}^{R} A_K^r = A_K$.

Por lo tanto, e^* se reduce a la forma

$$e^{\bullet}: \prod_{i=1}^{K-1} \left\{ \bigcap_{r=1}^{R} A_i^r \right\} \times \prod_{i=K}^{N} A_i = F.$$

Dado que este *edit* está en $\underline{e}_{K-1}$ y y lo falla sin importar el valor $y_K \in A_K$ que tome, entonces en particular y^0 lo falla. Tenemos así una contradicción. Se sigue de esto que al menos existe un valor $y_K \in A_K$ tal que $y = \left(y_1^0, ..., y_{K-1}^0, y_K, y_{K+1}^0, ..., y_N^0\right)$ satisface todos los *edits* en $\underline{e}_K$. ■

Ejemplo 2.3:

Consideremos los *edits* en el Ejemplo 2.1, a saber:

$e_1 : \langle A_1, \{0-14\}, A_3, A_4, \text{Alguna vez casado} \rangle = F$

$e_2 : \langle A_1, A_2, A_3, \{\text{Cónyuge}\}, \text{No está casado actualmente} \rangle = F$

$e^{\bullet} : \langle A_1, \{0-14\}, A_3, \{\text{Cónyuge}\}, A_5 \rangle = F$

Supongamos que tenemos el registro:

Sexo :	Masculino
Edad :	14
Educación :	Secundaria
Relación con el jefe de familia :	Otro
Estado civil :	Casado

Observamos que el registro falla el *edit* e_1 y pasa e_2 y $e^{\bullet}$.

Entonces, tomando K = 5, como el registro satisface los *edits* en $\underline{e}_4 = \{e^{\bullet}\}$ el Teorema 2.1 garantiza que existe al menos un valor para Estado civil que, junto con los valores de Sexo, Edad, Educación y Relación con el jefe de familia, harán que el registro satisfaga todos los *edits* en $\underline{e}_5$ (e.g., Estado civil = Soltero).

Corolario 2.1:

Suponga que un cuestionario tiene N campos y que un registro y^p satisface todos los *edits* en $\underline{e}_{K-1}$ pero no así algún *edit* en $\underline{e}_K$. Entonces existen valores y_i^0 $(i = K, ..., N)$ que harán que el registro $y^p{}' = (y_1^p, y_2^p, ..., y_{K-1}^p, y_K^0, y_{K+1}^0, ..., y_N^0)$ satisfaga todos los *edits*.

Demostración

Aplicando sucesivamente el Teorema 2.1 se llega al resultado deseado.■

Corolario 2.2:

Suponga que un registro tiene N campos con valores y_i $(i=1,...,N)$ y que S es un subconjunto de estos campos con la propiedad de que al menos uno de los valores y_i $(i\in S)$ aparece en cada *edit* fallado. Entonces existen valores y_i^0 $(i\in S)$ tales que el registro imputado que consiste de los valores y_i $(i\notin S)$, junto con los y_i^0 $(i\in S)$, satisface todos los *edits*.

Demostración

Reordenando los campos, ubicamos al conjunto S de campos como los últimos campos de los *edits* y del registro. Aplicando el Corolario 2.1 se llega al resultado deseado.■

Tenemos que, si seleccionamos el conjunto con el número mínimo de campos, entonces el Corolario 2.2 establece que al cambiar los valores en estos campos (manteniendo los valores en los otros campos sin cambiar) satisfaremos todos los *edits*.

Para garantizar que las aplicaciones repetidas del método dado por el Lema 2.1 generarán todos los *edits* implícitos esencialmente nuevos no redundantes, tenemos el siguiente teorema:

Teorema 2.2:

Si $e_p : \prod_{i=1}^{N} A_i^p = F$ es un *edit* implícito esencialmente nuevo no redundante, entonces puede ser generado por el proceso dado por el Lema 2.1.

Demostración

El hecho de que $e_p : \prod_{i=1}^{N} A_i^p = F$ sea implícito significa que todo registro que falle este *edit*, fallará también al menos uno de los *edits* explícitos. Considere los valores

$$y_i^0 \in A_i^p, \quad i=1,...,N-1, \tag{1}$$

lo que es posible dado que los A_i^p no son vacíos. Para cada valor $y_K \in A_N^p$, considere el registro $y_K^0 = \left(y_1^0, y_2^0, ..., y_{N-1}^0, y_K\right)$.

Se sigue de (1) que el registro $y_K^0 = \left(y_1^0, y_2^0, ..., y_K\right)$ debe fallar al menos uno de los *edits* explícitos para cada $K \in \left\{1, ..., n_N^p\right\}$. Supongamos que hay $R_N \in \left\{1, ..., \min\left\{n_N^p, \#\left(\underline{e}_C\right)\right\}\right\}$ *edits* explícitos con forma normal

$$e_{r_N} : \prod_{i=1}^{N} A_i^{r_N} = F, \quad r_N \in \left\{1, ..., R_N\right\}.$$

Considere ahora el siguiente *edit* implícito, con los anteriores como *edits* contribuyentes y campo generador *N*, dado por

$$e_q : \prod_{i=1}^{N} A_i^q = F,$$

donde $A_i^q = \bigcap_{r_N=1}^{R_N} A_i^{r_N}$, $i \in \left\{1, ..., N-1\right\}$ y $A_N^q = \bigcup_{r_N=1}^{R_N} A_N^{r_N}$.

Éste es un *edit* implícito ya que ninguna de las intersecciones es vacía $(y_i^0 \in A_i^q, \quad i \in \left\{1, ..., N-1\right\})$.

Ahora, $A_N^p \subseteq A_N^q$ ya que los *edits* fueron construidos para asegurar que cada $y_N \in A_N^p$ estuviera incluido en uno de los R_N *edits* y, así, en uno de los conjuntos $A_N^{r_N}$.

Dado que el valor y_{N-1}^0 fue arbitrariamente escogido del conjunto de valores A_{N-1}^p, un *edit* como e_q podría ser obtenido para cada valor $y_{N-1} \in A_{N-1}^p$. Tenemos así,

$$\begin{aligned} e_{r_{N-1}}^* : \prod_{i=1}^{N} A_i^{r_{N-1}} = F, \quad & r_{N-1} = 1, ..., R_{N-1}, \\ & R_{N-1} \in \left\{1, ..., \min\left\{n_{N-1}^p, \#\left(\underline{e}_{N-1}\right)\right\}\right\} \end{aligned} \tag{2}$$

donde, de acuerdo a la construcción de los *edits* $e_{r_{N-1}}$, tenemos que

$$A_N^{r_{N-1}} \supseteq A_N^p, \quad \forall\, r_{N-1} \in \left\{1, ..., R_{N-1}\right\}. \tag{3}$$

Considere el *edit* implícito con (2) como *edits* contribuyentes y campo generador $N-1$,

$$e_s : \prod_{i=1}^{N} A_i^s = F,$$

donde $A_i^s = \bigcap_{r_{N-1}=1}^{R_{N-1}} A_i^{r_{N-1}}$, $i \in \{1,...,N-2,N\}$ y $A_{N-1}^s = \bigcup_{r_{N-1}=1}^{R_{N-1}} A_{N-1}^{r_{N-1}}$.

Debido a la construcción de este *edit* tenemos que

$$A_{N-1}^s \supseteq A_{N-1}^p,$$

y debido a (3) tenemos que

$$A_N^s \supseteq A_N^p.$$

Continuando de esta forma con $N-2, N-3,...,1$, podemos generar un *edit* implícito

$$e_x : \prod_{j=1}^{N} A_j^x = F,$$

donde $A_j^x \supseteq A_j^p$, $\forall\, j \in \{1,...,N\}$.

Claramente, este *edit* es fallado por todo registro que falle el *edit* original del Teorema 2.2. Pero por hipótesis e_p no es redundante, es decir, no existe ningún *edit* implícito diferente tal que $A_j^p \subseteq A_j^x$ para alguna $j \in \{1,...,N\}$. Por lo tanto, $e_p = e_x$. ■

Hay que hacer notar que en el proceso dado por el Lema 2.1 se pueden obtener *edits* de la forma siguiente:

$$e_r : A_1 \times \cdots \times A_{j-1} \times A_j^r \times A_{j+1} \times \cdots \times A_N = F \qquad \text{con } A_j^r \subset A_j \text{ (subconjunto propio)}$$

Los cuales, aunque no son contradictorios en si mismos, contradicen el hecho de que A_j correspondía al conjunto de todos los valores permisibles en el campo j.

Un algoritmo sencillo permitiría a una computadora detectar este tipo de *edits* y eliminarlos.

Si tenemos un conjunto de *edits* que al aplicarles el Lema 2.1 nos arrojan un *edit* de la forma anterior, lo llamaremos un ***conjunto de edits inconsistentes*** y deberá ser analizado.

2.2 CONVERSIÓN A TEORÍA MATRICIAL.

Presentaremos en esta sección métodos prácticos para la conversión de los resultados de la sección anterior.

Una vez que los *edits* explícitos se han puesto en su forma matricial, un nuevo *edit* se obtiene siguiendo los pasos:

1. Se selecciona un campo generador y un conjunto de *edits* contribuyentes.
2. Para el campo generador se pone 1 en una entrada del campo si alguno de los *edits* contribuyentes tiene 1 en dicha entrada y 0 en otro caso.
3. Para los demás campos se pone 0 en una entrada del campo si alguno de los *edits* contribuyentes tiene 0 en dicha entrada y 1 en otro caso.

Este nuevo *edit* es válido a menos que alguno de los campos contenga 0's en todas sus entradas.

Ejemplo 2.4:

Para ilustrar esto retomamos el Ejemplo 1.13.

Derivemos un *edit* implícito de e_2 y e_3 usando Estado civil como el campo generador. Entonces la construcción del *edit* se representa mediante la siguiente tabla.

Edit	**Sexo**	**Edad**	**Educación**	**Relación con el jefe de familia**	**Estado civil**
e_2	1 1	1 0 0	1 1 1 1	1 1 1 1	0 1 1 1 1
e_3	1 1	1 1 1	1 1 1 1	1 1 0 0	1 0 1 1 1
	y	y	y	y	O
Edit implícito	1 1	1 0 0	1 1 1 1	1 1 0 0	1 1 1 1 1

Observamos que este nuevo *edit* es en realidad e_4.

Las siguientes reglas podrían formar la base de un algoritmo apropiado para generar el conjunto completo de *edits*:

1. Para producir un nuevo *edit* a partir de un conjunto cualquiera de *edits* éstos deben tener un campo en común en el que se vean involucrados (todos deben tener al menos un 0 entre los valores de dicho campo).

2. No se producirá un *edit* esencialmente nuevo de un *edit* implícito y un subconjunto de *edits* de los que el implícito haya surgido.

3. Una combinación de *edits* usando un campo en particular como generador no debe ser considerado si algún subconjunto de la combinación ha dado ya un *edit* implícito con el mismo campo generador.

Recordemos que un nuevo *edit* que sólo involucra a un campo indica que los *edits* explícitos son inconsistentes.

2.3 TEORÍA EN *EDITS* ARITMÉTICOS Y SU CONVERSIÓN A TEORÍA MATRICIAL.

Nos restringiremos, como en la sección 1.2, a *edits* lineales de la forma:

$$f(a_1,...,a_N) \geq 0,$$

donde f es una función lineal de las variables $a_1,...,a_N$ y la desigualdad será estricta de acuerdo a la especificación del *edit* original. Cualquier registro en el que $f \geq 0$ falla este *edit*.

Es fácil ver que los *edits* de este tipo también incluyen a los dos tipos más comunes de *edits* aritméticos: donde una identidad lineal debe ser satisfecha y donde la expresión lineal debe estar dentro de un rango de constantes.

De forma similar, una expresión racional de los campos dentro de un rango de constantes puede ser transformada a *edits* de esta forma. Es claro que, multiplicando por más o menos, un *edit* lineal puede ser también estandarizado siempre en la dirección de la desigualdad ya mostrada.

En esta sección nos restringiremos a *edits* lineales. Entonces cada *edit* se describe unívocamente con $N+1$ coeficientes y un indicador del tipo de desigualdad ($\geq$ ó $>$).

Tenemos entonces para cada *edit* una tabla como la siguiente:

Edit	Constante	Campo 1	Campo 2	...	Campo N	Indicador
e_r	α_0^r	α_1^r	α_2^r	...	α_N^r	δ^r

donde: $\alpha_i^r = 0$ si el *edit* no involucra al campo *i*

$$\delta^r = \begin{cases} 1 \text{ si la desigualdad es } > \\ 0 \text{ si la desigualdad es } \geq \end{cases}$$

Para propósitos de estandarización el primer coeficiente distinto de cero deberá ser ± 1.

La obtención de un conjunto completo de *edits* aritméticos se obtiene mediante el siguiente teorema.

Teorema 2.3:

Un *edit* implícito esencialmente nuevo e_t es generado por los *edits* e_r y e_s usando el campo *i* como generador, si y sólo si, α_i^r y α_i^s son diferentes de cero y de signo contrario. Entonces los coeficientes del nuevo *edit* son:

$$\alpha_k^t = \alpha_k^s \alpha_i^r - \alpha_k^r \alpha_i^s,\ k = 0,...,N \text{ y } \delta^t = \delta^r \delta^s.$$

Estos coeficientes deberán ser estandarizados haciendo el primer coeficiente diferente de 0 igual a ± 1.

Demostración

Tenemos que los *edits* e_r y e_s son de la forma

$$e_r : f(y_1,...,y_N) \geq 0$$
$$e_s : g(y_1,...,y_N) \geq 0$$

donde f y g son funciones lineales, es decir,

$$\begin{aligned} e_r &: \alpha_0^r + \alpha_1^r y_1 + \alpha_2^r y_2 + \cdots + \alpha_N^r y_N \geq 0, \\ e_s &: \alpha_0^s + \alpha_1^s y_1 + \alpha_2^s y_2 + \cdots + \alpha_N^s y_N \geq 0, \end{aligned} \quad \text{donde } \alpha_K^r, \alpha_K^s \in \mathbf{R} \quad \forall K \in \{0,...,N\}.$$

Supongamos, sin pérdida de generalidad, que el campo generador es N y que $\alpha_N^r > 0$ y $\alpha_N^s < 0$. Entonces

$$e_r : \beta_0^r + \beta_1^r y_1 + \beta_2^r y_2 + \cdots + \beta_{N-1}^r y_{N-1} + y_N \geq 0$$
$$e_s : \beta_0^s + \beta_1^s y_1 + \beta_2^s y_2 + \cdots + \beta_{N-1}^s y_{N-1} + y_N \leq 0$$

donde $\beta_i^r = \alpha_i^r / \alpha_N^r$, $\beta_i^s = \alpha_i^s / \alpha_N^s$, $\forall i \in \{0,...,N-1\}$. Así,

$$e_r : y_N \geq -\beta_0^r - \beta_1^r y_1 - \beta_2^r y_2 - \cdots - \beta_{N-1}^r y_{N-1}$$
$$e_s : y_N \leq -\beta_0^s - \beta_1^s y_1 - \beta_2^s y_2 - \cdots - \beta_{N-1}^s y_{N-1}$$

Supongamos ahora que tenemos un registro dado $y^0 = \left(y_1^0, y_2^0, ..., y_N^0\right)$ tal que

$$-\beta_0^r - \beta_1^r y_1^0 - \beta_2^r y_2^0 - \cdots - \beta_{N-1}^r y_{N-1}^0 \leq -\beta_0^s - \beta_1^s y_1^0 - \beta_2^s y_2^0 - \cdots - \beta_{N-1}^s y_{N-1}^0 \qquad (1)$$

En este caso, sin importar el valor del registro en el campo *N*, fallará al menos uno de los *edits* e_r ó e_s. Es más, en este caso tendremos que

$$y_N^0 \geq -\beta_0^s - \beta_1^s y_1^0 - \beta_2^s y_2^0 - \cdots - \beta_{N-1}^s y_{N-1}^0 \quad \text{ó}$$
$$y_N^0 \leq -\beta_0^r - \beta_1^r y_1^0 - \beta_2^r y_2^0 - \cdots - \beta_{N-1}^r y_{N-1}^0 \quad \text{ó}$$
$$-\beta_0^s - \beta_1^s y_1^0 - \beta_2^s y_2^0 - \cdots - \beta_{N-1}^s y_{N-1}^0 > y_N^0 > -\beta_0^r - \beta_1^r y_1^0 - \beta_2^r y_2^0 - \cdots - \beta_{N-1}^r y_{N-1}^0$$

En cualquiera de estos casos al menos e_r ó e_s fallarán. Por ejemplo, si $y_N^0 \geq -\beta_0^s - \beta_1^s y_1^0 - \beta_2^s y_2^0 - \cdots - \beta_{N-1}^s y_{N-1}^0$, entonces debido a (1) tenemos también que $y_N^0 \geq -\beta_0^r - \beta_1^r y_1^0 - \beta_2^r y_2^0 - \cdots - \beta_{N-1}^r y_{N-1}^0$, es decir, e_r falla.

Así, (1) es un *edit* implícito que puede ser reescrito como

$$(\beta_0^s - \beta_0^r) + (\beta_1^s - \beta_1^r) y_1 + (\beta_2^s - \beta_2^r) y_2 + \cdots + (\beta_{N-1}^s - \beta_{N-1}^r) y_{N-1} \leq 0$$

pero $\beta_i^s - \beta_i^r = \dfrac{\alpha_i^s}{\alpha_N^s} - \dfrac{\alpha_i^r}{\alpha_N^r} = \dfrac{\alpha_i^s \alpha_N^r - \alpha_i^r \alpha_N^s}{\alpha_N^s \alpha_N^r}$, $\forall i \in \{0, ..., N-1\}$. Como $\alpha_N^s \alpha_N^r < 0$, multiplicando por esta cantidad tenemos el *edit* implícito

$$e_t : \alpha_0^t + \alpha_1^t y_1 + \alpha_2^t y_2 + \cdots + \alpha_{N-1}^t y_{N-1} \geq 0$$

donde $\alpha_i^t = \alpha_i^s \alpha_N^r - \alpha_i^r \alpha_N^s$, $i \in \{0, ..., N-1\}$.

Es fácil verificar que la desigualdad estricta se cumple cuando e_r y e_s involucran ambas desigualdades estrictas.

Sin pérdida de generalidad, supongamos que $\alpha_N^r = 0$, entonces los *edits* contribuyentes quedan de la forma

$$e_r : \alpha_0^r + \alpha_1^r y_1 + \alpha_2^r y_2 + \cdots + \alpha_{N-1}^r y_{N-1} \geq 0,$$
$$e_s : \alpha_0^s + \alpha_1^s y_1 + \alpha_2^s y_2 + \cdots + \alpha_N^s y_N \geq 0$$

Entonces no podemos generar un *edit* implícito usando el campo N como generador pues éste no está presente en uno de los *edits* contribuyentes.

Basta mostrar que si α_N^r y α_N^s son del mismo signo (pero ambas distintas de cero), entonces ningún *edit* esencialmente nuevo puede ser implicado de e_r y e_s usando este procedimiento.

Sin perdida de generalidad, supongamos que α_N^r y α_N^s son ambas positivas. Obtenemos, análogamente,

$$e_r : y_N \geq -\beta_0^r - \beta_1^r y_1 - \beta_2^r y_2 - \cdots - \beta_{N-1}^r y_{N-1}$$
$$e_s : y_N \geq -\beta_0^s - \beta_1^s y_1 - \beta_2^s y_2 - \cdots - \beta_{N-1}^s y_{N-1}.$$

Supongamos que $e_t : \alpha_0^t + \alpha_1^t y_1 + \alpha_2^t y_2 + \cdots + \alpha_{N-1}^t y_{N-1} \geq 0$ es un *edit* implícito esencialmente nuevo generado por e_r y e_s. Sean $y_1^0, y_2^0, \ldots, y_{N-1}^0$ valores que fallen e_t, es decir, $\alpha_0^t + \alpha_1^t y_1^0 + \alpha_2^t y_2^0 + \cdots + \alpha_{N-1}^t y_{N-1}^0 \geq 0$.

Si e_r y e_s son *edits* diferentes, estos valores pueden también ser escogidos de forma que

$$-\beta_0^r - \beta_1^r y_1^0 - \beta_2^r y_2^0 - \cdots - \beta_{N-1}^r y_{N-1}^0 > -\beta_0^s - \beta_1^s y_1^0 - \beta_2^s y_2^0 - \cdots - \beta_{N-1}^s y_{N-1}^0.$$

Ahora sea y_N^0 un valor tal que

$$-\beta_0^r - \beta_1^r y_1^0 - \beta_2^r y_2^0 - \cdots - \beta_{N-1}^r y_{N-1}^0 > -\beta_0^s - \beta_1^s y_1^0 - \beta_2^s y_2^0 - \cdots - \beta_{N-1}^s y_{N-1}^0 > y_N^0.$$

Claramente, el registro $y^0 = \left(y_1^0, y_2^0, \ldots, y_N^0\right)$ fallará e_t, además satisfará e_r y e_s. Así e_t no puede ser un *edit* implícito generado por e_r y e_s (utilizando a N como campo generador) si α_N^r y α_N^s son del mismo signo.■

De hecho, el Teorema 2.3 establece que de dos desigualdades lineales donde los signos de la desigualdad están en la misma dirección, una variable puede ser eliminada tomando sus combinaciones lineales si y sólo si la variable tiene coeficientes en las dos desigualdades con signos opuestos. También se puede mostrar que en el caso de los *edits* aritméticos, las aplicaciones repetidas del Teorema 2.3 derivarán en todos los *edits* implícitos esencialmente nuevos. Para obtener el conjunto de *edits* implícitos basta combinar los *edits* apropiados usando como campos generadores todos los posibles campos de igual manera que con los *edits* lógicos.

Es importante recalcar que la teoría hasta ahora desarrollada para las variables continuas se aplica únicamente para relaciones lineales entre ellas, la cual es una desventaja que deberá tomarse en cuenta cuando se considere la implementación de esta metodología en la validación e imputación de una encuesta específica.

CAPÍTULO III
MÉTODOS DE IMPUTACIÓN SECUENCIAL Y CONJUNTA

En este capítulo desarrollamos algoritmos que, dado un registro, identificarán un conjunto de campos (el de menor cardinalidad) cuyos valores puedan ser cambiados de tal forma que el registro resultante satisfaga todos los *edits*.

Nos restringiremos a métodos de imputación del tipo *hot deck*. Cuando en un registro debemos imputar el valor de un campo, estos métodos imputan dicho valor de un registro que pasó todos los *edits*. Métodos de este tipo tratan de mantener la distribución de los datos dada por los registros que anteriormente pasaron los *edits*.

Ya que los métodos requieren que conozcamos el conjunto mínimo de campos a imputar, la siguiente sección permitirá su identificación.

3.1 ALGORITMO FH PARA LA IDENTIFICACIÓN DEL CONJUNTO MÍNIMO DE CAMPOS A IMPUTAR.

Si algún *edit* falló un registro en particular, será necesario identificar el conjunto mínimo de campos a imputar. Supondremos que contamos con el conjunto completo de *edits* (Sección 1.3).

Consideremos una matriz $R' \times N$ donde R' es el número de *edits* fallados del conjunto completo y N el número de campos en el cuestionario. La entrada (r,n) de la matriz será 0 a menos que el campo n esté involucrado en el *edit* r, en cuyo caso será 1.

Ejemplo 3.1:

Consideremos la siguiente tabla:

Edit	**Campo**				
	1	2	3	. . .	N
e_2	1	1	0	. . .	0
.	.	.	.		.
.	.	.	.		.
.	.	.	.		.
e_R	0	1	1	. . .	1

Entonces la tabla indica que e_1 es satisfecho y por lo tanto no aparece en la matriz, e_2 es fallado e involucra a los campos 1 y 2, e_R es fallado e involucra los campos 2, 3, ..., N.

A esta matriz la llamaremos la ***matriz de edits fallados***.

Entonces nuestro problema se reduce a encontrar el conjunto más pequeño de campos (columnas) que juntos tengan al menos 1 en cada campo fallado (fila).

Un procedimiento para realizar esto es dado por el siguiente algoritmo:

*Algoritmo FH**:

1. Ignorar todos los *edits* satisfechos (filas que contengan únicamente ceros).

2. Identificar todos los campos que fallan *edits* simples. Por definición, éstos deben ser incluidos en el conjunto mínimo. Generamos así una matriz modificada eliminando los *edits* en los que intervienen estos campos. Si se eliminaron todos los *edits* el conjunto mínimo ha sido identificado.

3. Si el paso 2 no eliminó todos los *edits*, identificar el *edit* que involucre al menor número de campos (escoger aleatoriamente si hay más de uno). Al menos uno de los campos involucrados en este *edit* debe estar en el conjunto mínimo.

4. Para cada uno de estos campos, generar una matriz modificada como en el paso 2. Lleve un registro de las combinaciones de los campos hasta ahora seleccionados.

* Con base a las ideas dadas en I. P. Fellegi y D. Holt (1976).

5. Repita los pasos 3 y 4 hasta que la primera matriz modificada desaparezca.

Ejemplo 3.2:

Retomemos el Ejemplo 1.13.

Consideremos el siguiente registro:

Sexo:	Masculino
Edad:	12
Educación:	Media superior
Relación con el jefe de familia:	Esposa
Estado civil:	Divorciado

Entonces la matriz de *edits* fallados para este registro es:

Edit	**Sexo**	**Edad**	**Educación**	**Relación con el jefe de familia**	**Estado civil**
E_1	1	0	0	1	0
E_2	0	1	0	0	1
E_3	0	0	0	1	1
E_4	0	1	0	1	0
E_5	0	1	1	0	0

Por ejemplo, e_1 involucra a los campos Sexo y Relación con el jefe de familia, entonces ponemos 1's en estos dos campos y 0's en los tres restantes. Como no hay *edits* que se satisfagan ni *edits* simples que fallen, los pasos 1 y 2 no reducen el número de campos.

Comenzamos entonces el ciclo dado por los pasos 3 y 4.

Ciclo 1:

Seleccionaremos un *edit* que involucre el mínimo número de campos. En este caso todos los *edits* involucran dos campos; entonces, arbitrariamente, seleccionamos e_1. Así, en el conjunto mínimo tendremos Sexo o Relación con el jefe de familia.

Eliminamos todos los *edits* que involucren al Sexo o a la Relación con el jefe de familia. Generamos dos matrices modificadas: una para Sexo y otra para Relación con el jefe de familia.

Ciclo 2:

Si seleccionamos Sexo para imputar en el Ciclo 1 obtenemos la siguiente tabla.

Edit	Edad	Educación	Relación con el jefe de familia	Estado civil
e_2	1	0	0	1
e_3	0	0	1	1
e_4	1	0	1	0
e_5	1	1	0	0

Seleccionando e_2 (arbitrariamente, dado que todos los *edits* involucran el mismo número de campos) podríamos imputar Edad o Estado civil.

Los posibles campos del conjunto mínimo son:

Sexo y Edad
Sexo y Estado civil

Si seleccionamos Relación con el jefe de familia en el Ciclo 1 obtenemos la siguiente tabla.

Edit	Sexo	Edad	Educación	Estado civil
e_2	0	1	0	1
e_5	0	1	1	0

Seleccionando e_5 (arbitrariamente) obtendríamos los siguientes posibles campos en el conjunto mínimo:

Relación con el jefe de familia y Edad
Relación con el jefe de familia y Educación

Que, junto con los dos anteriores, corresponden a los cuatro posibles conjuntos de campos en el conjunto mínimo después del ciclo 2. Obtenemos así las siguientes cuatro matrices modificadas de *edits* fallados.

Edit	Educación	Relación con el jefe de familia	Estado civil
e_3	0	1	1

Matriz de edits *fallados si Sexo y Edad son imputados*

Edit	Edad	Educación	Relación con el jefe de familia
e_4	1	0	1
e_5	1	1	0

Matriz de edits *fallados si Sexo y Estado civil son imputados*

Si Relación con el jefe de familia y Edad son imputados, obtenemos una matriz de *edits* fallados vacía.

Edit	Sexo	Edad	Estado civil
e_2	0	1	1

Matriz de edits *fallados si Relación con el jefe de familia y Educación son imputados*

Ya que al final del Ciclo 2 obtuvimos una matriz de *edits* fallados vacía no necesitamos continuar: los campos en el conjunto mínimo son Relación con el jefe de familia y Edad. Por supuesto en este sencillo ejemplo el resultado podría ser verificado directamente de la primer matriz de *edits* fallados: la Relación con el jefe de familia y Edad cubren juntos todos los *edits* fallados, pero no lo hacen otros pares de campos.

Otros acercamientos podrían ser desarrollados para solucionar el problema de forma más eficiente; el procedimiento propuesto es una iteración relativamente sencilla para identificar todos los posibles conjuntos mínimos para propósitos de imputación.

3.2 MÉTODO SECUENCIAL DE IMPUTACIÓN.

Supondremos en esta sección que tenemos un registro $\{y_i^0\}_{i=1}^N$ donde los primeros K campos van a ser imputados, siendo estos el conjunto mínimo de campos a cambiar. Tendremos además un conjunto completo de *edits* obtenido en el capítulo anterior.

El algoritmo imputará primero el campo K y después sistemáticamente los campos $K-1, K-2, ..., 1$. Sea M el número de *edits* en los cuales el campo K está específicamente involucrado pero no lo están los campos $1, 2, ..., K-1$. Sean éstos:

$$e_r : \prod_{i=1}^{N} A_i^r = F \quad r = 1, ..., M \quad \text{con} \quad A_K^r \subset A_K. \tag{3.1}$$

Para un registro dado separamos aquellos *edits* en (3.1) para los cuales $y_i^0 \notin A_i^r$, para al menos uno de los $i = K, K+1, ..., N$.

De los *M edits* consideramos los *M' edits* para los que

$$y_i^0 \in A_i^r \quad \forall i = K+1,...,N;\ r = 1,...,M' \text{ con } M' \leq M.$$

Los *edits* que cumplen esto son aquellos para los que un registro fallará o pasará dependiendo únicamente de su valor en el campo *K*. Ahora, si queremos satisfacer todos estos *edits* imputando un valor en el campo *K*, dicho valor, y_K^*, debe satisfacer

$$y_K^* \in \bigcup_{r=1}^{M'} (A_K^r)^c.$$

Esto es siempre posible ya que, si la unión de conjuntos de la derecha fuera vacío, de acuerdo al Teorema 2.1 significaría que hay un *edit* que el registro falla y que involucra únicamente a los campos $K+1, K+2,..., N$. Lo cual sería una contradicción, pues entonces habría que imputar otro campo aparte de los *K* primeros, pero entonces los primeros *K* campos no serían el conjunto mínimo de campos a cambiar.

Entonces habiendo imputado y_K^* se satisfacen todos los *edits* que involucran al campo *K* y alguno de los campos $K+1, K+2,..., N$ (pero no a los campos $1,2,3,...,K-1$). Consideraremos ahora todos los *edits* que tienen que ver con los campos $K-1,..., N$ pero no con los campos $1,2,...,K-2$. Y así sucesivamente hasta que todos los campos $1,2,...,K$ hayan sido imputados y, por construcción, todos los *edits* se satisfagan.

Ejemplo 3.3:

Retomemos el Ejemplo 1.13. Tenemos el conjunto completo de *edits*:

$$e_1 : \langle \{\text{Masculino}\}, A_2, A_3, \{\text{Esposa}\}, A_5 \rangle = F$$
$$e_2 : \langle A_1, \{0-14\}, A_3, A_4, \text{Alguna vez casado} \rangle = F$$
$$e_3 : \langle A_1, A_2, A_3, \{\{\text{Esposa}\}, \{\text{Esposo}\}\}, \text{No está casado actualmente} \rangle = F$$
$$e_4 : \langle A_1, \{0-14\}, A_3, \{\{\text{Esposa}\}, \{\text{Esposo}\}\}, A_5 \rangle = F$$
$$e_5 : \langle A_1, \{\{0-14\}, \{15-16\}\}, \{\text{Media superior}\}, A_4, A_5 \rangle = F$$

Se da un registro que contiene las siguientes características:

Sexo :	Masculino
Edad :	12
Educación :	Primaria
Relación con el jefe de familia :	Esposa
Estado civil :	Casado

Es fácil verificar que los *edits* e_1, e_2 y e_4 son fallados por el registro. Necesitamos localizar el conjunto mínimo de campos que cubra todos los *edits*. Se tiene que un único campo no cubre todos estos *edits*, existen tres pares de campos que lo hacen y son: Sexo y Edad, Edad y Relación con el jefe de familia y Relación con el jefe de familia y Estado civil.

Supongamos que se ha tomado la decisión de imputar Sexo y Edad, dejando los otros tres campos sin cambios (entonces $K = 2$). Ahora, consideremos todos los *edits* que involucren Edad (la K-ésima variable), pero que no involucren Sexo (la variable $K-1=1$); éstos son los *edits* e_2, e_4 y e_5. Entonces $M = 3$. Sin embargo, e_5 se satisface sin tomar en cuenta los valores de los campos $K+1, K+2, K+3$ (los tres campos que no son imputados). Como Educación ≠ Media superior, e_5 se satisfará sin importar el valor que imputemos en Edad, así, e_5 no necesita ser considerado cuando imputemos Edad (por lo tanto, $M' = 2$). Así, tenemos que considerar a los conjuntos $(A_2^2)^c$ y $(A_2^4)^c$ (los complementos del conjunto de valores para Edad en los *edits* e_2 y e_4). Por lo que utilizando el Método 1, tenemos que escoger un valor para la Edad y_2^* tal que

$$y_2^* \in (A_2^2)^c \cup (A_2^4)^c = \{\{15-16\},\{17+\}\} \cup \{\{15-16\},\{17+\}\} = \{\{15-16\},\{17+\}\}$$

Así debemos imputar una edad mayor o igual a 15 años. Habiendo imputado Edad (supongamos 22 años), imputaremos Sexo. Consideraremos otra vez todos los *edits* que involucren Sexo (no hay otros campos que imputar). Sólo el *edit* e_1 lo involucra. Verificamos si e_1 se satisface sin tomar en cuenta los valores de los campos ya imputados o sin cambios; éste no es el caso. Así ahora $M = M' = 1$. Tenemos que imputar un valor y_1^* para Sexo tal que:

$$y_1^* \in (A_1^1)^c = \{\text{Femenino}\}$$

Entonces, la única imputación factible es Sexo = Femenino.

Si se da el caso en que, después de imputar $K-m+1$ campos, no hay *edits* fallados que tengan que ver con el campo m sin que estén involucrados los campos $m-1, m-2, \ldots, 1$, entonces imputamos cualquier valor y continuamos con el campo $m-1$.

Desventajas del Método:

1. No se toma en cuenta otra información del registro que podría asociarse al campo a imputar.

2. Como los campos se imputan uno por uno, la distribución conjunta de los valores imputados en los campos individuales será diferente a la de la población, aunque se mantengan las distribuciones marginales.

Posibles Mejoras al Método:

Se puede restringir la búsqueda a los registros que tengan otros campos con valores idénticos a los de nuestro registro.

3.3 ALGORITMO SECUENCIAL DE IMPUTACIÓN.

Traduciremos ahora el método de imputación propuesto en la sección anterior a la representación desarrollada en las secciones de conversión matricial (Secciones 1.4 y 2.2).

Suponga que tenemos N campos, los K primeros a imputar en un registro.

Algoritmo Secuencial de Imputación

1. Identifique los *edits* que se satisfacen tomando en cuenta los valores en los campos $K+1,...,N$ sin importar los valores en los campos $1,...,K$.

2. Considere aquellos *edits* que involucran al campo K y no a los campos $1,...,K-1$. Se deberá imputar un valor que satisfaga estos *edits*. Considere la submatriz de los valores del campo K y los últimos *edits* obtenidos. Agregue un nuevo renglón y ponga un 1 si todo renglón tiene un 1 en ese valor y un 0 en otro caso.

3. Busque aleatoriamente de entre los registros anteriormente aceptados aquel que posea uno de los posibles valores del campo K e impute su valor.

4. Repita los pasos 1, 2 y 3 ahora con el campo $K-1$ y así sucesivamente hasta agotar los campos a imputar.

Ejemplo 3.4:

Éste está en términos del ejemplo anterior.

Supongamos que imputamos primero Edad (Campo 2), y después Sexo (Campo 1).

1. Identificamos los *edits* que toman en cuenta sólo los valores de los otros 3 campos, sin importar lo que imputemos en Edad y Sexo. Observamos en la Tabla 1.1 las columnas para las que el registro tiene 1's en los 3 campos que no vamos a imputar. Buscamos en estas 3 columnas los *edits* que tengan 0's en alguna columna. Es fácil ver que estos *edits* serán e_3 y e_5. Entonces e_1, e_2 y e_4 (el complemento de los anteriores) pueden posiblemente imponer restricciones a la imputación de Edad y Sexo.

2. Consideramos de entre los *edits* e_1, e_2 y e_4 aquellos que involucran Edad pero no Sexo. Estos son e_2 y e_4 (pues tienen sólo 1's en el campo Sexo, ver Tabla

1.1). Así que tenemos que imputar Edad de tal forma que estos dos *edits* sean satisfechos. El Teorema 2.1 asegura que esto puede hacerse siempre. Observando la parte Edad de e_2 y e_4, obtenemos la tabla:

Edit	0-14	15-16	17+
e_2	1	0	0
e_4	1	0	0

Si combinamos columna por columna los valores de tal forma que la fila resultante contenga 1 cuando todas las filas contengan 1 y contenga 0 en otro caso, obtendremos en este caso simple la representación de posibles imputaciones para Edad:

0-14	15-16	17+
1	0	0

Ahora, si imputamos para Edad algún valor donde la tabla previa contenga 0, obtendremos que los *edits* e_2 y e_4 son satisfechos. De esta forma las imputaciones para Edad son 15-16 ó 17+.

3. Buscamos de entre los registros previamente aceptados uno con uno de los valores de Edad anteriores. Imputamos el valor de Edad de ese registro.

4. En este sencillo ejemplo ahora sólo podemos imputar la variable Sexo y de entre los *edits* e_1, e_2 y e_4 el que involucra Sexo es e_1. Como el único valor posible para Sexo que permite que e_1 sea satisfecho es Femenino, ésta es la imputación.

5. Repetimos el paso 3 para la variable Sexo.

3.4 MÉTODO DE IMPUTACIÓN CONJUNTA.

A diferencia del primer método, en éste consideraremos, a partir del conjunto mínimo, los M'' *edits* cuya falla dependa de los valores de los campos $1,...,K$, es decir, los *edits*

$$e_r : \prod_{i=1}^{N} A_i^r = F \qquad \text{donde } y_i^0 \in A_i^r \quad \forall i = K+1,...,N;\ r = 1,...,M''.$$

Considere los conjuntos:

$$A_i^* = \bigcap_{r=1}^{M^*} A_i^r \quad \forall i = K+1,...,N.$$

Observemos que $A_i^* \neq \varnothing$ ya que $y_i^0 \in A_i^r, \quad r = 1,...,M''$.

Si escogemos un registro procesado previamente cuyos valores en los campos $K+1,...,N$ están en el conjunto A_i^*, entonces, dado que este registro satisface todos los *edits*, sus valores en dichos campos pueden ser usados para la imputación de nuestro registro y satisfará automáticamente todos los *edits*.

Ejemplo 3.5:

Utilizaremos el Ejemplo 3.3. Hay dos campos a imputar: Sexo y Edad. Consideramos todos los *edits* que involucran estos campos, a saber: e_1, e_2, e_4, e_5. De éstos, e_5 se satisfará sin importar qué valor imputemos para Sexo y Edad porque el valor en Educación es Primaria, así que sólo se deben de considerar e_1, e_2 y e_4, es decir, $M'' = 3$.

Los conjuntos A_i^* son:

$$A_3^* = A_3^1 \cap A_3^2 \cap A_3^4 = A_3$$
$$A_4^* = A_4^1 \cap A_4^2 \cap A_4^4 = \{\text{Esposa}\} \cap A_4 \cap \{\{\text{Esposa}\},\{\text{Esposo}\}\} = \{\text{Esposa}\}$$
$$A_5^* = A_5^1 \cap A_5^2 \cap A_5^4 = A_5 \cap \text{Alguna vez casado} \cap A_5 = \text{Alguna vez casado}$$

Así que si buscamos entre los registros previamente procesados, hasta que encontramos uno para el cual Educación sea cualquier valor, Relación con el jefe de familia sea Esposa y Estado civil sea alguno dentro del subconjunto Alguna vez casado, entonces podemos imputar sus valores de Sexo y Edad a nuestro registro.

Ventajas del Método:

1. Sólo se realiza la búsqueda de un registro para imputar sus valores (en lugar de los K registros del método anterior).

2. Al hacerse una imputación conjunta no sólo el valor en cada campo aparecerá en la misma proporción que la de la población, sino que también la incidencia de combinaciones de valores.

3. Se elimina la posibilidad de que en un registro se imputen dos o más campos cuyos valores casi nunca se presentan juntos en un registro.

Desventajas del Método:

Dado que el subconjunto de la población del que obtendremos el registro es más pequeño que el del método secuencial, puede tomar más tiempo su búsqueda.

Posibles Mejoras al Método:

El método puede fácilmente extenderse para que tome la información de otros campos de nuestro registro y los incorpore a la búsqueda del registro a imputar. De hecho, si el número de campos y el de valores para cada campo no es muy grande, se puede eliminar el cálculo de los conjuntos A_i^* y sólo tomar los valores a imputar de un registro procesado previamente cuyos campos relevantes sean los mismos que los de nuestro registro.

La implementación de estos métodos depende de la complejidad de los *edits*, del número de campos y de sus posibles valores, así como de la frecuencia de la falla de un *edit*. Es casi seguro que la implementación del Método de Imputación Conjunta necesitará un método de *default* en caso de que no se pueda encontrar un registro imputable en un período de tiempo razonable, por ejemplo, el Método Secuencial de Imputación.

3.5 ALGORITMO DE IMPUTACIÓN CONJUNTA.

Traduciremos ahora el método de imputación propuesto en la sección anterior a la representación desarrollada en las secciones de conversión matricial.

Suponga que tenemos N campos, los K primeros a imputar en un registro.

Algoritmo de Imputación Conjunta

1. Identifique los *edits* que se satisfacen tomando en cuenta los valores en los campos $K+1,...,N$ sin importar los valores en los campos $1,...,K$.

2. Considere los campos que no van a ser imputados $(K+1,...,N)$. Para cada uno, identifique los valores que contienen 1 en los *edits* obtenidos del paso anterior.

3. Busque aleatoriamente de entre los registros anteriormente aceptados aquel que en los campos $K+1,...,N$ tiene cualquier combinación de los valores obtenidos en el paso anterior. Impute los valores $1,...,K$ de ese registro.

Ejemplo 3.6:

Retomamos el Ejemplo 3.3.

1. Conservamos de entre todos los *edits* a e_1, e_2 y e_4 como en el paso 1 del Ejemplo 3.5.

2. Consideramos los campos que no van a ser imputados: Estado civil, Relación con el jefe de familia y Educación. Los valores que contienen 1 en cada uno de los *edits* son:

Educación:	Cualquier valor
Relación con el jefe de familia:	Esposa
Estado civil:	Casado, Divorciado, Separado, Viudo

3. Supongamos que al buscar de entre los anteriores registros aceptados, encontramos uno para el cual los valores para los campos que no van a ser imputados son:

Educación:	Secundaria
Relación con el jefe de familia:	Esposa
Estado civil:	Separado

Y para ese registro:

Edad:	22
Sexo:	Femenino

Entonces imputamos en nuestro registro Edad = 22 y Sexo = Femenino.

Será fácil modificar estos métodos para utilizar información en el cuestionario que no esté relacionada explícitamente en los *edits* con los campos que requieren imputación. El problema consiste en limitar la búsqueda de registros previamente procesados a aquellos que tengan valores particulares en campos particulares, determinando los valores o rangos de valores aceptables del registro a imputar.

Ejemplo 3.7:

En el ejemplo anterior la imputación de Sexo y Edad no se vio restringida por la Educación. Sin embargo, podríamos, como una limitante adicional, insistir que siempre que la Edad se impute pero no la Educación, el registro que utilicemos para imputar deberá tener el mismo valor para Educación, que el de nuestro registro.

Hay que hacer notar que, mientras más limitantes pongamos al registro del que vamos a imputar, más larga será la búsqueda de dicho registro de entre los registros previamente procesados.

CAPÍTULO IV

EL PROBLEMA DEL *SET COVERING* EN LA GENERACIÓN DEL CONJUNTO COMPLETO DE *EDITS*

Podemos observar en los capítulos anteriores que no se ha proporcionado un método sistemático para obtener los *edits* implícitos. Es con este fin que se han buscado métodos para obtenerlos. Robert S. Garfinkel y George L. Nemhauser (1972) presentan el problema del *set covering* (PSC), el cual permite la obtención del conjunto de *edits* implícitos a partir de los *edits* explícitos de una forma sistemática.

En la primera parte de este capítulo presentamos el PSC en su versión general dado por Garfinkel y Nemhauser (1972) y en la segunda parte presentamos la aplicación del PSC en la generación de *edits* implícitos así como un ejemplo ilustrativo.

4.1. EL PROBLEMA DEL *SET COVERING.*

Consideremos un conjunto $I = \{1,...,m\}$ y un conjunto $P = \{P_1,...,P_n\}$, donde $P_j \subseteq I$, $\forall j \in J = \{1,...,n\}$.

Un subconjunto $J^* \subseteq J$ define una ***cubierta*** de I si:

$$\bigcup_{j \in J^*} P_j = I$$

Y, si además, $\forall j,k \in J^*$ con $j \neq k$, $P_j \cap P_k = \varnothing$, J^* define una ***partición*** de I. Por conveniencia nos referiremos al conjunto J^* como a una cubierta o partición.

Sea un costo $c_j > 0$ asociado a cada $j \in J$. El costo total de la cubierta J^* es $\sum_{j \in J^*} c_j$. El ***problema del set covering*** (PSC) consiste en encontrar una cubierta de mínimo costo y puede ser escrito como:

$$\text{Minimizar } x_0 = \sum_{j=1}^{n} c_j x_j \tag{4.1}$$

$$\text{Sujeto a } \sum_{j=1}^{n} a_{ij} x_j \geq 1, \quad \forall i \in \{1,...,m\} \tag{4.2}$$

$$x_j = 0,1 \ \forall j \in \{1,...,n\} \tag{4.3}$$

donde:

$$x_j = \begin{cases} 1, \text{ si } j \text{ está en la cubierta} \\ 0, \text{ en otro caso} \end{cases}$$

$$a_{ij} = \begin{cases} 1, \text{ si } i \in P_j \\ 0, \text{ en otro caso} \end{cases}$$

Similarmente, el ***problema del set partitioning*** es obtenido al reemplazar (4.2) con:

$$\sum_{j=1}^{n} a_{ij} x_j = 1, \ \forall i \in \{1,...,m\} \tag{4.2a}$$

Cualquier x que satisfaga (4.2) y (4.3) ó (4.2a) y (4.3) es llamada una ***cubierta*** o ***partición solución***, respectivamente.

Para cualquier cubierta J, un elemento $j^* \in J$ se dice que será ***redundante*** si $J - \{j^*\}$ es también una cubierta. Si una cubierta contiene más de un elemento redundante, es también llamada redundante. Una cubierta que no es redundante es llamada ***cubierta principal***. Y se le llama ***cubierta principal solución*** a la solución de una cubierta principal.

4.2. APLICACIÓN DEL PROBLEMA DEL *SET COVERING* A LA GENERACIÓN DEL CONJUNTO COMPLETO DE EDITS.

El Lema 2.1 requiere que fijemos un campo $i \in \{1,...,n\}$, denominado campo generador. Por lo tanto, tendremos un PSC para cada campo i. Tenemos el conjunto $A_i = \{0,...,n_i - 1\}$

de los posibles valores del campo i y sean $\underline{e}_i = \{e_r\}_{r \in S_i}$ con $S_i \subset \{1,...,m\}$, el conjunto de los *edits* que entran en el campo i y $P_i = \{A_i^r\}_{r \in S_i}$.

Como en la Sección 4.1, un subconjunto $S_i^* \subseteq S_i$ define una cubierta de A_i si:

$$\bigcup_{r \in S_i^*} A_i^r = A_i.$$

Y, si además, $\forall r,t \in S_i^*$ con $r \neq t$, $A_i^r \cap A_i^t = \varnothing$, S_i^* define una partición de A_i.

El PSC para la generación de *edits* implícitos consiste en encontrar una cubierta. Cabe aclarar que en este caso las c_j's, es decir los costos, son iguales a uno, por lo que el PSC puede ser escrito como:

$$\text{Minimizar} \sum_{r \in S_i} x_r \tag{4.4}$$

$$\text{Sujeto a} \sum_{r \in S_i} g_{jr}^i x_r \geq 1, \quad \forall j \in A_i \tag{4.5}$$

$$x_r = 0,1 \ \forall r \in S_i \tag{4.6}$$

donde:

$$x_r = \begin{cases} 1, \text{ si } r \text{ está en la cubierta} \\ 0, \text{ en otro caso} \end{cases}$$

$$g_{jr}^i = \begin{cases} 1, \text{ si el valor } j \in A_i \text{ entra en el edit } e_r \\ 0, \text{ en otro caso} \end{cases}$$

Cualquier x que satisfaga (4.5) y (4.6) es llamada una ***cubierta solución***. Una x que satisfaga

$$\sum_{r \in S_i} g_{jr}^i x_r = 1 \quad \forall j \in A_i \tag{4.5a}$$

y (4.6) es llamada una ***partición solución***.

Una vez obtenida la cubierta principal solución para cada cubierta principal de este PSC (las cuales son vectores de 0's y 1's) generamos un *edit* implícito por cada cubierta principal solución considerando como *edits* contribuyentes a los 1's de ésta y como campo generador a i, siguiendo el procedimiento dado por el Lema 2.1.

En caso de que se generen *edits* implícitos esencialmente nuevos no redundantes, éstos se anexarán al conjunto de *edits* explícitos a considerarse en el siguiente PSC con campo fijo

$i+1$. Tenemos así un ciclo que consistirá en n PSC's, uno por cada campo. Este ciclo deberá repetirse hasta que ya no se obtengan más *edits* implícitos esencialmente nuevos no redundantes.

Ejemplo 4.1:

Supongamos que tenemos 6 campos con:

$$A_1=\{0,1\} \quad n_1=2 \qquad A_4=\{0,1,2,3\} \quad n_4=4$$
$$A_2=\{0,1,2\} \quad n_2=3 \qquad A_5=\{0,1,2\} \quad n_5=3$$
$$A_3=\{0,1\} \quad n_3=2 \qquad A_6=\{0,1,2,3\} \quad n_6=4$$

Consideremos los siguientes *edits* explícitos:

$$e_1:\langle A_1,\{0,1\},\{0\},A_4,\{0,1\},A_6\rangle=F,$$
$$e_2:\langle\{1\},A_2,\{1\},\{0,1\},A_5,\{2,3\}\rangle=F,$$
$$e_3:\langle\{0\},\{1,2\},A_3,\{1,2,3\},A_5,A_6\rangle=F,$$
$$e_4:\langle A_1,\{0,2\},A_3,A_4,A_5,\{0,1\}\rangle=F,$$
$$e_5:\langle\{1\},A_2,A_3,\{0\},\{1,2\},A_6\rangle=F.$$

Tomemos el campo 1 como generador.

Tenemos que $A_1=\{0,1\}$ es el conjunto de valores que puede tomar el campo 1. El conjunto de *edits* donde entra el campo 1 es $\underline{e}_1=\{e_2,e_3,e_5\}$ y el conjunto de sus subíndices es $S_1=\{2,3,5\}$. De esta manera, $A_1^2=\{1\}, A_1^3=\{0\}$ y $A_1^5=\{1\}$.

Ahora, para obtener las cubiertas principales solución, necesitamos obtener la matriz $G_1=(g_{jr}^1)$ y asociarle su respectivo PSC ya que al resolverlo aseguramos que obtenemos todas las cubiertas principales solución necesarias para generar los *edits* implícitos esencialmente nuevos.

De lo anterior tenemos que:

$$g^1{}_{02}=0 \qquad g^1{}_{03}=1 \qquad g^1{}_{05}=0$$
$$g^1{}_{12}=1 \qquad g^1{}_{13}=0 \qquad g^1{}_{15}=1$$

Así, la matriz G_1 está dada por:

$$G_1 = \begin{array}{c} \\ 0 \\ 1 \end{array} \begin{array}{c} \begin{array}{ccc} e_2 & e_3 & e_5 \end{array} \\ \begin{pmatrix} 0 & 1 & 0 \\ 1 & 0 & 1 \end{pmatrix} \end{array}$$

Entonces, el PSC asociado a la matriz G_1 es:

$$\begin{array}{lllll} \text{Minimizar} & x_2 + & x_3 + & x_5 & \\ \text{Sujeto a} & & x_3 & & \geq 1 \\ & x_2 + & & x_5 & \geq 1 \\ & x_2, x_3, x_5 = 0, 1 & & & \end{array}$$

Por lo tanto, sus cubiertas principales solución son:

$$(1 \;\; 1 \;\; 0), (0 \;\; 1 \;\; 1).$$

Una vez obtenidas las cubiertas principales solución (donde su primer elemento corresponde a e_2, el segundo a e_3 y el tercero a e_5), se generan los *edits* implícitos esencialmente nuevos, aplicando el Lema 2.1 como a continuación se muestra.

$$(1 \;\; 1 \;\; 0) \leftrightarrow e^*\left(1, \{e_2, e_3\}\right) : \langle A_1, \{1,2\}, \{1\}, \{1\}, A_5, \{2,3\}\rangle = F$$
$$(0 \;\; 1 \;\; 1) \leftrightarrow e^*\left(1, \{e_3, e_5\}\right) : \langle A_1, \{1,2\}, A_3, \varnothing, \{1,2\}, A_6\rangle = F$$

De los cuales, el único válido y esencialmente nuevo es $e^*\left(1, \{e_2, e_3\}\right)$ al no ser vacío en ninguno de sus campos y tener el campo 1 completo. Además, al no dominar ni ser dominado es un *edit* implícito esencialmente nuevo no redundante que denotaremos e_6.

Tomemos el campo 2 como generador.

Tenemos que $A_2 = \{0,1,2\}$, $\underline{e}_2 = \{e_1, e_3, e_4, e_6\}$, $S_2 = \{1,3,4,6\}$ y $A_2^1 = \{0,1\}$, $A_2^3 = \{1,2\}$, $A_2^4 = \{0,2\}$, $A_2^6 = \{1,2\}$.

La matriz G_2 está dada por:

$$G_2 = \begin{array}{c} \\ 0 \\ 1 \\ 2 \end{array} \begin{array}{c} \begin{array}{cccc} e_1 & e_3 & e_4 & e_6 \end{array} \\ \begin{pmatrix} 1 & 0 & 1 & 0 \\ 1 & 1 & 0 & 1 \\ 0 & 1 & 1 & 1 \end{pmatrix} \end{array}$$

Entonces, el PSC asociado a la matriz G_2 es:

$$\begin{array}{lllll} \text{Minimizar} & x_1 + x_3 + x_4 + x_6 & & \\ \text{Sujeto a} & x_1 + \quad\; x_4 & & \geq 1 \\ & x_1 + x_3 + \quad\; x_6 & & \geq 1 \\ & \quad\; x_3 + x_4 + x_6 & & \geq 1 \\ & x_1, x_3, x_4, x_6 = 0, 1 & & \end{array}$$

Por lo tanto, sus cubiertas principales solución son:

$$(1\ 1\ 0\ 0),(1\ 0\ 1\ 0),(1\ 0\ 0\ 1),(0\ 1\ 1\ 0),(0\ 0\ 1\ 1).$$

Generamos los *edits* implícitos a partir de estas cubiertas usando el Lema 2.1:

$$(1\ 1\ 0\ 0) \leftrightarrow e^*(2,\{e_1,e_3\}) : \langle \{0\}, A_2, \{0\}, \{1,2,3\}, \{0,1\}, A_6 \rangle = F$$
$$(1\ 0\ 1\ 0) \leftrightarrow e^*(2,\{e_1,e_4\}) : \langle A_1, A_2, \{0\}, A_4, \{0,1\}, \{0,1\} \rangle = F$$
$$(1\ 0\ 0\ 1) \leftrightarrow e^*(2,\{e_1,e_6\}) : \langle A_1, A_2, \varnothing, \{1\}, \{0,1\}, \{2,3\} \rangle = F$$
$$(0\ 1\ 1\ 0) \leftrightarrow e^*(2,\{e_3,e_4\}) : \langle \{0\}, A_2, A_3, \{1,2,3\}, A_5, \{0,1\} \rangle = F$$
$$(0\ 0\ 1\ 1) \leftrightarrow e^*(2,\{e_4,e_6\}) : \langle A_1, A_2, \{1\}, \{1\}, A_5, \varnothing \rangle = F$$

De los cuales, los *edits* esencialmente nuevos no redundantes son $e^*(2,\{e_1,e_3\})$, $e^*(2,\{e_1,e_4\})$, $e^*(2,\{e_3,e_4\})$ y los denotaremos e_7, e_8 y e_9, respectivamente.

Tomemos el campo 3 como generador.

Tenemos que $A_3 = \{0,1\}$, $\underline{e}_3 = \{e_1, e_2, e_6, e_7, e_8\}$, $S_3 = \{1,2,6,7,8\}$ y $A_3^1 = \{0\}$, $A_3^2 = \{1\}, A_3^6 = \{1\}, A_3^7 = \{0\}, A_3^8 = \{0\}$.

La matriz G_3 está dada por:

$$G_3 = \begin{array}{c} \\ 0 \\ 1 \end{array} \begin{array}{c} \begin{array}{ccccc} e_1 & e_2 & e_6 & e_7 & e_8 \end{array} \\ \begin{pmatrix} 1 & 0 & 0 & 1 & 1 \\ 0 & 1 & 1 & 0 & 0 \end{pmatrix} \end{array}$$

Entonces, el PSC asociado a la matriz G_3 es:

$$\begin{array}{lllll} \text{Minimizar} & x_1 + x_2 + x_6 + x_7 + x_8 & & \\ \text{Sujeto a} & x_1 + \quad\; x_7 + x_8 & & \geq 1 \\ & \quad\; x_2 + x_6 & & \geq 1 \\ & x_1, x_2, x_6, x_7, x_8 = 0, 1 & & \end{array}$$

Por lo tanto, sus cubiertas principales solución son:

$$(1\ 1\ 0\ 0\ 0),(1\ 0\ 1\ 0\ 0),(0\ 1\ 0\ 1\ 0),(0\ 1\ 0\ 0\ 1),$$
$$(0\ 0\ 1\ 1\ 0),(0\ 0\ 1\ 0\ 1).$$

Generamos los *edits* implícitos a partir de estas cubiertas usando el Lema 2.1:

$$(1\ 1\ 0\ 0\ 0) \leftrightarrow e^*(3,\{e_1,e_2\}):\langle\{1\},\{0,1\},A_3,\{0,1\},\{0,1\},\{2,3\}\rangle = F$$
$$(1\ 0\ 1\ 0\ 0) \leftrightarrow e^*(3,\{e_1,e_6\}):\langle A_1,\{1\},A_3,\{1\},\{0,1\},\{2,3\}\rangle = F$$
$$(0\ 1\ 0\ 1\ 0) \leftrightarrow e^*(3,\{e_2,e_8\}):\langle\{1\},A_2,A_3,\{0,1\},\{0,1\},\varnothing\rangle = F$$
$$(0\ 1\ 0\ 0\ 1) \leftrightarrow e^*(3,\{e_2,e_7\}):\langle\varnothing,A_2,A_3,\{1\},\{0,1\},\{2,3\}\rangle = F$$
$$(0\ 0\ 1\ 1\ 0) \leftrightarrow e^*(3,\{e_6,e_8\}):\langle A_1,\{1,2\},A_3,\{1\},\{0,1\},\varnothing\rangle = F$$
$$(0\ 0\ 1\ 0\ 1) \leftrightarrow e^*(3,\{e_6,e_7\}):\langle\{0\},\{1,2\},A_3,\{1\},\{0,1\},\{2,3\}\rangle = F$$

De los cuales, los *edits* esencialmente nuevos no redundantes son $e^*(3,\{e_1,e_2\})$ y $e^*(3,\{e_1,e_6\})$ que denotaremos e_{10} y e_{11}, respectivamente.

Tomemos el campo 4 como generador.

Tenemos que $A_4=\{0,1,2,3\}$, $\underline{e}_4=\{e_2,e_3,e_5,e_6,e_7,e_9,e_{10},e_{11}\}$,
$S_4=\{2,3,5,6,7,9,10,11\}$ y $A_4^2=\{0,1\}, A_4^3=\{1,2,3\}, A_4^5=\{0\}, A_4^6=\{1\}$,
$A_4^7=\{1,2,3\}, A_4^9=\{1,2,3\}, A_4^{10}=\{0,1\}, A_4^{11}=\{1\}$.

La matriz G_4 está dada por:

$$G_4 = \begin{array}{c} \\ 0 \\ 1 \\ 2 \\ 3 \end{array} \begin{array}{c} \begin{array}{cccccccc} e_2 & e_3 & e_5 & e_6 & e_7 & e_9 & e_{10} & e_{11} \end{array} \\ \begin{pmatrix} 1 & 0 & 1 & 0 & 0 & 0 & 1 & 0 \\ 1 & 1 & 0 & 1 & 1 & 1 & 1 & 1 \\ 0 & 1 & 0 & 0 & 1 & 1 & 0 & 0 \\ 0 & 1 & 0 & 0 & 1 & 1 & 0 & 0 \end{pmatrix} \end{array}$$

Entonces, el PSC asociado a la matriz G_4 es:

$$\begin{array}{lcccccccccccc} \text{Minimizar} & x_2 & + & x_3 & + & x_5 & + & x_6 & + & x_7 & + & x_9 & + & x_{10} & + & x_{11} & \\ \text{Sujeto a} & x_2 & + & & & x_5 & + & & & & & & & x_{10} & & & \geq 1 \\ & x_2 & + & x_3 & + & & & x_6 & + & x_7 & + & x_9 & + & x_{10} & + & x_{11} & \geq 1 \\ & & & x_3 & + & & & & & x_7 & + & x_9 & & & & & \geq 1 \\ & & & & & x_2, x_3, x_5, x_6, x_7, x_9, x_{10}, x_{11} = 0, 1 & & & & & & & & & & & \end{array}$$

Por lo tanto, sus cubiertas principales solución son:

$$(1\ 1\ 0\ 0\ 0\ 0\ 0\ 0),(1\ 0\ 0\ 0\ 0\ 1\ 0\ 0),$$
$$(1\ 0\ 0\ 0\ 0\ 0\ 1\ 0),(0\ 1\ 1\ 0\ 0\ 0\ 0\ 0),$$
$$(0\ 1\ 0\ 0\ 0\ 0\ 0\ 1),(0\ 0\ 1\ 0\ 0\ 1\ 0\ 0),$$
$$(0\ 0\ 1\ 0\ 0\ 0\ 1\ 0),(0\ 0\ 0\ 0\ 0\ 1\ 0\ 1),$$
$$(0\ 0\ 0\ 0\ 0\ 0\ 1\ 1).$$

Generamos los *edits* implícitos a partir de estas cubiertas usando el Lema 2.1:

$$(1\ 1\ 0\ 0\ 0\ 0\ 0\ 0) \leftrightarrow e^*(4,\{e_2,e_3\}):\langle\varnothing,\{1,2\},\{1\},A_4,A_5,\{2,3\}\rangle = F$$
$$(1\ 0\ 0\ 0\ 0\ 1\ 0\ 0) \leftrightarrow e^*(4,\{e_2,e_7\}):\langle\varnothing,A_2,\varnothing,A_4,\{0,1\},\{2,3\}\rangle = F$$
$$(1\ 0\ 0\ 0\ 0\ 0\ 1\ 0) \leftrightarrow e^*(4,\{e_2,e_9\}):\langle\varnothing,A_2,\{1\},A_4,A_5,\varnothing\rangle = F$$
$$(0\ 1\ 1\ 0\ 0\ 0\ 0\ 0) \leftrightarrow e^*(4,\{e_3,e_5\}):\langle\varnothing,\{1,2\},A_3,A_4,\{1,2\},A_6\rangle = F$$
$$(0\ 1\ 0\ 0\ 0\ 0\ 0\ 1) \leftrightarrow e^*(4,\{e_3,e_{10}\}):\langle\varnothing,\{1\},A_3,A_4,\{0,1\},\{2,3\}\rangle = F$$
$$(0\ 0\ 1\ 0\ 0\ 1\ 0\ 0) \leftrightarrow e^*(4,\{e_5,e_7\}):\langle\varnothing,A_2,\{0\},A_4,\{1\},A_6\rangle = F$$
$$(0\ 0\ 1\ 0\ 0\ 0\ 1\ 0) \leftrightarrow e^*(4,\{e_5,e_9\}):\langle\varnothing,A_2,A_3,A_4,\{1,2\},\{0,1\}\rangle = F$$
$$(0\ 0\ 0\ 0\ 0\ 1\ 0\ 1) \leftrightarrow e^*(4,\{e_7,e_{10}\}):\langle\varnothing,\{0,1\},\{0\},A_4,\{0,1\},\{2,3\}\rangle = F$$
$$(0\ 0\ 0\ 0\ 0\ 0\ 1\ 1) \leftrightarrow e^*(4,\{e_9,e_{10}\}):\langle\varnothing,\{0,1\},A_3,A_4,\{0,1\},\varnothing\rangle = F$$

De los cuales, ninguno es un *edit* esencialmente nuevo no redundante.

Tomemos el campo 5 como generador.

Tenemos que $A_5 = \{0,1,2\}$, $\underline{e}_5 = \{e_1,e_5,e_7,e_8,e_{10},e_{11}\}$, $S_5 = \{1,5,7,8,10,11\}$ y $A_5^1 = \{0,1\}, A_5^5 = \{1,2\}, A_5^7 = \{0,1\}, A_5^8 = \{0,1\}, A_5^{10} = \{0,1\}, A_5^{11} = \{0,1\}$.

La matriz G_5 está dada por:

$$G_5 = \begin{array}{c} \\ 0 \\ 1 \\ 2 \end{array}\!\!\begin{array}{c} \begin{array}{cccccc} e_1 & e_5 & e_7 & e_8 & e_{10} & e_{11} \end{array} \\ \begin{pmatrix} 1 & 0 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 & 1 \\ 0 & 1 & 0 & 0 & 0 & 0 \end{pmatrix} \end{array}$$

Entonces, el PSC asociado para la matriz G_5 es:

$$
\begin{array}{lccccccccc}
\text{Minimizar} & x_1 & + x_5 & + x_7 & + x_8 & + x_{10} & + x_{11} & & \\
\text{Sujeto a} & x_1 & + & x_7 & + x_8 & + x_{10} & + x_{11} & \geq 1 \\
 & x_1 & + x_5 & + x_7 & + x_8 & + x_{10} & + x_{11} & \geq 1 \\
 & & x_5 & & & & & \geq 1 \\
 & & x_1, x_5, x_7, x_8, x_{10}, x_{11} = 0, 1 & & & & &
\end{array}
$$

Por lo tanto, sus cubiertas principales solución son:

$$(1\ 1\ 0\ 0\ 0\ 0),(0\ 1\ 1\ 0\ 0\ 0),(0\ 1\ 0\ 1\ 0\ 0),$$
$$(0\ 1\ 0\ 0\ 1\ 0),(0\ 1\ 0\ 0\ 0\ 1).$$

Generamos los *edits* implícitos a partir de estas cubiertas usando el Lema 2.1:

$$(1\ 1\ 0\ 0\ 0\ 0) \leftrightarrow e^*(5,\{e_1,e_5\}):\langle\{1\},\{0,1\},\{0\},\{0\},A_5,A_6\rangle = F$$
$$(0\ 1\ 1\ 0\ 0\ 0) \leftrightarrow e^*(5,\{e_5,e_7\}):\langle\varnothing,A_2,\{0\},\varnothing,A_5,A_6\rangle = F$$
$$(0\ 1\ 0\ 1\ 0\ 0) \leftrightarrow e^*(5,\{e_5,e_8\}):\langle\{1\},A_2,\{0\},\{0\},A_5,\{0,1\}\rangle = F$$
$$(0\ 1\ 0\ 0\ 1\ 0) \leftrightarrow e^*(5,\{e_5,e_{10}\}):\langle\{1\},\{0,1\},A_3,\{0\},A_5,\{2,3\}\rangle = F$$
$$(0\ 1\ 0\ 0\ 0\ 1) \leftrightarrow e^*(5,\{e_5,e_{11}\}):\langle\{1\},\{1\},A_3,\varnothing,A_5,\{2,3\}\rangle = F$$

De los cuales, los esencialmente nuevos no redundantes son $e^*(5,\{e_1,e_5\})$, $e^*(5,\{e_5,e_8\})$ y $e^*(5,\{e_5,e_{10}\})$ que denotaremos e_{12}, e_{13} y e_{14}, respectivamente.

Tomemos el campo 6 como generador.

Tenemos que $A_6 = \{0,1,2,3\}$, $\underline{e}_6 = \{e_2,e_4,e_6,e_8,e_9,e_{10}e_{11},e_{13},e_{14}\}$,
$S_6 = \{2,4,6,8,9,10,11,13,14\}$ y $A_6^2 = \{2,3\}, A_6^4 = \{0,1\}, A_6^6 = \{2,3\}, A_6^8 = \{0,1\}$,
$A_6^9 = \{0,1\}, A_6^{10} = \{2,3\}, A_6^{11} = \{2,3\}, A_6^{13} = \{0,1\}, A_6^{14} = \{2,3\}$.

La matriz G_6 está dada por:

$$
G_6 = \begin{array}{c} \\ 0 \\ 1 \\ 2 \\ 3 \end{array}
\begin{array}{c}
\begin{array}{ccccccccc} e_2 & e_4 & e_6 & e_8 & e_9 & e_{10} & e_{11} & e_{13} & e_{14} \end{array} \\
\begin{pmatrix}
0 & 1 & 0 & 1 & 1 & 0 & 0 & 1 & 0 \\
0 & 1 & 0 & 1 & 1 & 0 & 0 & 1 & 0 \\
1 & 0 & 1 & 0 & 0 & 1 & 1 & 0 & 1 \\
1 & 0 & 1 & 0 & 0 & 1 & 1 & 0 & 1
\end{pmatrix}
\end{array}
$$

Entonces, el PSC asociado para la matriz G_6 es:

$$
\begin{array}{lcccccccccccccccl}
\text{Minimizar} & x_2 & + & x_4 & + & x_6 & + & x_8 & + & x_9 & + & x_{10} & + & x_{11} & + & x_{13} & + & x_{14} & \\
\text{Sujeto a} & & & x_4 & + & & & x_8 & + & x_9 & + & & & & & x_{13} & & & \geq 1 \\
& x_2 & + & & & x_6 & + & & & & & x_{10} & + & x_{11} & + & & & x_{14} & \geq 1 \\
& & & & & & & x_2, x_4, x_6, x_8, x_9, x_{10}, x_{11}, x_{13}, x_{14} = 0, 1 & & & & & & & & & &
\end{array}
$$

Por lo tanto, sus cubiertas principales solución son:

$$
\begin{array}{l}
(1\ 1\ 0\ 0\ 0\ 0\ 0\ 0\ 0),(1\ 0\ 0\ 1\ 0\ 0\ 0\ 0\ 0), \\
(1\ 0\ 0\ 0\ 1\ 0\ 0\ 0\ 0),(1\ 0\ 0\ 0\ 0\ 0\ 0\ 1\ 0), \\
(0\ 1\ 1\ 0\ 0\ 0\ 0\ 0\ 0),(0\ 1\ 0\ 0\ 0\ 1\ 0\ 0\ 0), \\
(0\ 1\ 0\ 0\ 0\ 0\ 1\ 0\ 0),(0\ 1\ 0\ 0\ 0\ 0\ 0\ 0\ 1), \\
(0\ 0\ 0\ 1\ 0\ 0\ 1\ 0\ 0),(0\ 0\ 0\ 1\ 0\ 0\ 0\ 0\ 1), \\
(0\ 0\ 0\ 0\ 1\ 1\ 0\ 0\ 0),(0\ 0\ 0\ 0\ 1\ 0\ 1\ 0\ 0), \\
(0\ 0\ 0\ 0\ 1\ 0\ 0\ 0\ 1),(0\ 0\ 0\ 0\ 0\ 1\ 0\ 1\ 0), \\
(0\ 0\ 0\ 0\ 0\ 0\ 1\ 1\ 0),(0\ 0\ 0\ 0\ 0\ 0\ 0\ 1\ 1).
\end{array}
$$

Generamos los *edits* implícitos a partir de estas cubiertas usando el Lema 2.1:

$$
\begin{array}{l}
(0\ 1\ 0\ 0\ 0\ 0\ 1\ 0\ 0) \leftrightarrow e^*\left(6,\{e_4,e_{11}\}\right):\langle A_1,\varnothing,A_3,\{1\},\{0,1\},A_6\rangle = F \\
(0\ 1\ 0\ 0\ 0\ 0\ 0\ 0\ 1) \leftrightarrow e^*\left(6,\{e_4,e_{14}\}\right):\langle\{1\},\{0\},A_3,\{0\},A_5,A_6\rangle = F \\
(1\ 0\ 0\ 0\ 1\ 0\ 0\ 0\ 0) \leftrightarrow e^*\left(6,\{e_2,e_9\}\right):\langle\varnothing,A_2,\{1\},\{1\},A_5,A_6\rangle = F \\
(1\ 0\ 0\ 0\ 0\ 0\ 0\ 1\ 0) \leftrightarrow e^*\left(6,\{e_2,e_{13}\}\right):\langle\{1\},A_2,\varnothing,\{0\},A_5,A_6\rangle = F \\
(0\ 1\ 1\ 0\ 0\ 0\ 0\ 0\ 0) \leftrightarrow e^*\left(6,\{e_4,e_6\}\right):\langle A_1,\{2\},\{1\},\{1\},A_5,A_6\rangle = F \\
(0\ 1\ 0\ 0\ 0\ 1\ 0\ 0\ 0) \leftrightarrow e^*\left(6,\{e_4,e_{10}\}\right):\langle\{1\},\{0\},A_3,\{0,1\},\{0,1\},A_6\rangle = F \\
(0\ 0\ 1\ 0\ 1\ 0\ 0\ 0\ 0) \leftrightarrow e^*\left(6,\{e_6,e_9\}\right):\langle\{0\},\{1,2\},\{1\},\{1\},A_5,A_6\rangle = F \\
(0\ 0\ 1\ 0\ 0\ 0\ 0\ 1\ 0) \leftrightarrow e^*\left(6,\{e_6,e_{13}\}\right):\langle\{1\},\{1,2\},\varnothing,\varnothing,A_5,A_6\rangle = F \\
(0\ 0\ 0\ 1\ 0\ 1\ 0\ 0\ 0) \leftrightarrow e^*\left(6,\{e_8,e_{10}\}\right):\langle\{1\},\{0,1\},\{0\},\{0,1\},\{0,1\},A_6\rangle = F \\
(0\ 0\ 0\ 1\ 0\ 0\ 1\ 0\ 0) \leftrightarrow e^*\left(6,\{e_8,e_{11}\}\right):\langle A_1,\{1\},\{0\},\{1\},\{0,1\},A_6\rangle = F \\
(0\ 0\ 0\ 1\ 0\ 0\ 0\ 0\ 1) \leftrightarrow e^*\left(6,\{e_8,e_{14}\}\right):\langle\{1\},\{0,1\},\{0\},\{0\},\{0,1\},A_6\rangle = F \\
(0\ 0\ 0\ 0\ 1\ 1\ 0\ 0\ 0) \leftrightarrow e^*\left(6,\{e_9,e_{10}\}\right):\langle\varnothing,\{0,1\},A_3,\{1\},\{0,1\},A_6\rangle = F \\
(0\ 0\ 0\ 0\ 1\ 0\ 1\ 0\ 0) \leftrightarrow e^*\left(6,\{e_9,e_{11}\}\right):\langle\{0\},\{1\},A_3,\{1\},\{0,1\},A_6\rangle = F \\
(0\ 0\ 0\ 0\ 1\ 0\ 0\ 0\ 1) \leftrightarrow e^*\left(6,\{e_9,e_{14}\}\right):\langle\varnothing,\{0,1\},A_3,\varnothing,A_5,A_6\rangle = F
\end{array}
$$

$(0\ 0\ 0\ 0\ 0\ 1\ 0\ 1\ 0) \leftrightarrow e^*(6,\{e_{10},e_{13}\}):\langle\{1\},\{0,1\},\{0\},\{0\},\{0,1\},A_6\rangle = F$

$(0\ 0\ 0\ 0\ 0\ 0\ 1\ 1\ 0) \leftrightarrow e^*(6,\{e_{11},e_{13}\}):\langle\{1\},\{1\},\{0\},\varnothing,\{0,1\},A_6\rangle = F$

$(0\ 0\ 0\ 0\ 0\ 0\ 0\ 1\ 1) \leftrightarrow e^*(6,\{e_{13},e_{14}\}):\langle\{1\},\{0,1\},\{0\},\{0\},A_5,A_6\rangle = F$

De los cuales, los esencialmente nuevos no redundantes son $e^*(6,\{e_2,e_4\})$, $e^*(6,\{e_4,e_6\})$, $e^*(6,\{e_4,e_{10}\})$ y $e^*(6,\{e_4,e_{14}\})$ que denotaremos e_{15}, e_{16}, e_{17} y e_{18}, respectivamente.

Hemos completado un ciclo de seis PSC's. Se puede verificar que un nuevo ciclo no genera nuevos *edits* implícitos esencialmente nuevos no redundantes, por lo que nos detenemos en este punto.

La Tabla 4.1 resume todos los *edits* implícitos esencialmente nuevos no redundantes generados en este ejemplo, su campo generador y sus *edits* contribuyentes.

Campo generador	***Edits* contribuyentes**	***Edit* implícito esencialmente nuevo no redundante**
1	e_2, e_3	$e_6:\langle A_1,\{1,2\},\{1\},\{1\},A_5,\{2,3\}\rangle = F$
2	e_1, e_3	$e_7:\langle\{0\},A_2,\{0\},\{1,2,3\},\{0,1\},A_6\rangle = F$
	e_1, e_4	$e_8:\langle A_1,A_2,\{0\},A_4,\{0,1\},\{0,1\}\rangle = F$
	e_3, e_4	$e_9:\langle\{0\},A_2,A_3,\{1,2,3\},A_5,\{0,1\}\rangle = F$
3	e_1, e_2	$e_{10}:\langle\{1\},\{0,1\},A_3,\{0,1\},\{0,1\},\{2,3\}\rangle = F$
	e_1, e_6	$e_{11}:\langle A_1,\{1\},A_3,\{1\},\{0,1\},\{2,3\}\rangle = F$
5	e_1, e_5	$e_{12}:\langle\{1\},\{0,1\},\{0\},\{0\},A_5,A_6\rangle = F$
	e_5, e_8	$e_{13}:\langle\{1\},A_2,\{0\},\{0\},A_5,\{0,1\}\rangle = F$
	e_5, e_{10}	$e_{14}:\langle\{1\},\{0,1\},A_3,\{0\},A_5,\{2,3\}\rangle = F$
6	e_2, e_4	$e_{15}:\langle\{1\},\{0,2\},\{1\},\{0,1\},A_5,A_6\rangle = F$
	e_4, e_6	$e_{16}:\langle A_1,\{2\},\{1\},\{1\},A_5,A_6\rangle = F$
	e_4, e_{10}	$e_{17}:\langle\{1\},\{0\},A_3,\{0,1\},\{0,1\},A_6\rangle = F$
	e_4, e_{14}	$e_{18}:\langle\{1\},\{0\},A_3,\{0\},A_5,A_6\rangle = F$

Tabla 4.1. Edits *implícitos esencialmente nuevos no redundantes del Ejemplo 4.1.*

4.3. LA METODOLOGÍA DE FELLEGI Y HOLT COMO PROBLEMA DEL *SET COVERING.*

El objetivo de la metodología de Fellegi y Holt es, a partir de un registro que falla *edits*, producir un nuevo registro que no falle ningún *edit* en $\underline{e}_E$, cambiando el registro dado lo menos posible. Así, si el registro $y^0 \in \underline{E}_E$ (i.e., y^0 falla algún *edit*) es revisado, un modelo razonable sería:

$$\begin{aligned} &\text{Minimizar} \quad \sum_{j \in J} c_j x_j \\ &\text{Sujeto a} \quad y \in A - \underline{E}_E \\ &\qquad x_j = \begin{cases} 1 \text{ si } y_j \neq y_j^0 \\ 0 \text{ en otro caso} \end{cases} \quad \text{donde } j = 1,\ldots,n \end{aligned} \tag{4.7}$$

El coeficiente c_j en (4.7) es una medida de la "confiabilidad" del campo j, es decir, c_j es grande si el campo j es poco probable que tenga error. El modelo dado por (4.7) es llamado el de ***mínimos campos ponderados a imputar*** (MCPI) y si $c_j = 1$ para toda j, es llamado el de ***mínimos campos a imputar*** (MCI).

Considera el PSC:

$$\begin{aligned} &\text{Minimizar} \quad \sum_{j \in J} c_j x_j \\ &\text{Sujeto a} \quad \sum_{j \in J} a_{ij} x_j \geq 1, \\ &\qquad x_j = 0,1, \quad j \in J \end{aligned} \tag{4.8}$$

donde:

$$a_{ij} = \begin{cases} 1 \text{ si el campo } j \text{ entra en } e_i \in \underline{e}_F(y^0) \\ 0 \text{ en otro caso} \end{cases} \quad \text{y}$$

$\underline{e}_F(y^0)$ es el conjunto de *edits* fallados por y^0.

Sería razonable postular que una cubierta solución para (4.8) también resuelve (4.7) en el sentido de que especifica qué campos hay que cambiar. Claramente, cualquier $x = (x_1,\ldots,x_n)$ factible en (4.7) deberá satisfacer las restricciones de (4.8), las cuales establecen que para cada *edit* fallado al menos un campo que entre en él deberá ser cambiado. Para ver que una x puede ser factible para (4.8) pero no para (4.7) consideremos el siguiente ejemplo:

Ejemplo 4.2:

Retomando el Ejemplo 4.1, consideremos el registro $(1 \quad 0 \quad 0 \quad 0 \quad 1 \quad 0)$, que falla e_1, e_4 y e_5. El PSC (4.8) tiene restricciones:

$$\begin{aligned} x_2 + x_3 + x_5 &\geq 1 \\ x_2 + x_6 &\geq 1 \\ x_1 + x_4 + x_5 &\geq 1 \\ x_1, \ldots, x_6 &= 0, 1 \end{aligned}$$

Una cubierta solución del PSC es $x = (0 \quad 1 \quad 0 \quad 0 \quad 1 \quad 0)$, pero ningún cambio posible en sólo los campos 2 y 5 lleva a un registro factible.

I. P. Fellegi y D. Holt (1976) señalan esta situación y un resultado clave es que, si el conjunto de *edits* es expandido para que incluya el conjunto de los *edits* implícitos, entonces (4.8) corresponde a (4.7) en el sentido de que los campos óptimos a imputar son identificados (aun cuando la imputación todavía debe determinarse).

Fundamentalmente, la importancia de los *edits* implícitos esencialmente nuevos no redundantes es que ellos generan restricciones de PSC potencialmente fuertes pues $a_{\bullet g} = 0$ en (4.8) (es decir, el campo generador g no entra en el *edit* implícito esencialmente nuevo $e^{\bullet}$).

CAPÍTULO V
ALGORITMOS GKL

La metodología de Fellegi y Holt para validar e imputar un registro y^0 consiste de dos pasos:

a) Generar el conjunto completo de *edits* ($\underline{e}_C$).

b) Si $y^0 \in \underline{E}_E$ (i.e., y^0 falla algún *edit*), resolver (4.8) para determinar los campos a imputar considerando que $\underline{e}_F(y^0)$ son todos los *edits* en $\underline{e}_C$ fallados por y^0.

Cada paso requiere la solución de un problema potencialmente difícil. El primer paso requiere la generación de un conjunto de *edits* que razonablemente puede esperarse que crezca exponencialmente con relación al número de *edits* explícitos. El segundo paso requiere la solución de un PSC bastante grande para cada registro fallado.

I. P. Fellegi y D. Holt (1976) no presentan experiencia computacional y tratan en forma muy básica el segundo problema. También hicieron notar que en la generación de *edits* se deberían tomar todos los posibles subconjuntos tanto de *edits* explícitos como implícitos que se fueran generando como *edits* contribuyentes y todos los campos como campo generador, pero que se podía lograr cierta eficiencia usando las siguientes observaciones:

1) Los conjuntos de *edits* pueden producir *edits* esencialmente nuevos sólo si tienen al menos un campo involucrado en común.

2) Si $e^*(i,\{e_r\}_{r\in S})$ es esencialmente nuevo, entonces no hay necesidad de generar $e^*(i,\{e_r\}_{r\in S} \cup e^*(i,\{e_r\}_{r\in S}))$ ya que no será esencialmente nuevo.

3) Si $e^*(i,\{e_r\}_{r\in S'})$ es esencialmente nuevo y $S' \subseteq S$, entonces $e^*(i,\{e_r\}_{r\in S})$ será dominado por $e^*(i,\{e_r\}_{r\in S'})$.

Mientras que 1) – 3) pueden tener cierto valor en el proceso de generación de *edits*, no proveen un ahorro substancial.

En este capítulo daremos un algoritmo para reducir los cálculos necesarios para obtener el conjunto completo de *edits*. Un segundo algoritmo permitirá imputar un registro particular sin la necesidad de dicho conjunto.

5.1 RESULTADOS TEÓRICOS.

Los siguientes resultados nos darán las bases teóricas para los algoritmos presentados en las siguientes secciones.

Definamos los siguientes conjuntos de *edits*: $\underline{e}^*(i) = \{e^*(i,\underline{e}) \mid \underline{e} \subseteq \underline{e}_E\}$ y $\underline{e}^*(i,j) = \{e^*(j,\underline{e}) \mid \underline{e} \subseteq \underline{e}_E \cup \underline{e}^*(i)\}$, donde $\underline{e}_E$ es el conjunto de *edits* explícitos. Así, $\underline{e}^*(i,j)$ es el conjunto de *edits* implícitos que pueden ser producidos usando como campos generadores al campo i y después al campo j.

Primero, consideremos el siguiente lema:

Lema 5.1:

Sea $\underline{e}_Q = \underline{e}_E \cup \underline{e}^*(i) \cup \underline{e}^*(j)$ y $\underline{e}^*(j,\underline{e}_Q) = \left\{e^*(j,\underline{e}) \mid \underline{e} \subset \underline{e}^*(i) \cup \underline{e}_E, \underline{e} \cap \underline{e}^*(i) \neq \varnothing\right\}$.
Entonces $\underline{e}^*(i,j) = \underline{e}_Q \cup \underline{e}^*(j,\underline{e}_Q)$ y $\underline{e}^*(j,i) = \underline{e}_Q \cup \underline{e}^*(i,\underline{e}_Q)$.

Demostración

Primero observemos que $\underline{e}_Q \cap \underline{e}^*(j,\underline{e}_Q) = \varnothing$, ya que en la generación de un *edit* en $\underline{e}^*(j,\underline{e}_Q)$ interviene al menos un *edit* en $\underline{e}^*(i)$ usando el campo j como generador.

La demostración de que $\underline{e}^*(i,j) = \underline{e}_Q \cup \underline{e}^*(j,\underline{e}_Q)$ se hará por contenciones.

Sea $e_x \in \underline{e}^*(i,j)$, entonces existe $\underline{e}_x \subseteq \underline{e}_E \cup \underline{e}^*(i)$ tal que $e_x = e^*(j,\underline{e}_x)$. Tenemos así los siguientes casos.

1) Si $\underline{e}_x = \varnothing$, entonces $E_x = \varnothing$ y por lo tanto, $e_x \in \underline{e}_Q \cup \underline{e}^*(j,\underline{e}_Q)$.
2) Si $\underline{e}_x = \{e_E\} \in \underline{e}_E$, entonces $e_x = e_E$ y por lo tanto, $e_x \in \underline{e}_E \subseteq \underline{e}_Q \subseteq \underline{e}_Q \cup \underline{e}^*(j,\underline{e}_Q)$.
3) Si $\underline{e}_x = \{e^*(i)\} \in \underline{e}^*(i)$, entonces $e_x = e^*(i)$ y por lo tanto, $e_x \in \underline{e}^*(i) \subseteq \underline{e}_Q \subseteq \underline{e}_Q \cup \underline{e}^*(j,\underline{e}_Q)$.
4) Si $\underline{e}_x \subseteq \underline{e}_E$ tiene más de un *edit*, entonces $e_x \in \underline{e}^*(j) \subseteq \underline{e}_Q \subseteq \underline{e}_Q \cup \underline{e}^*(j,\underline{e}_Q)$.
5) Si $\underline{e}_x \subseteq \underline{e}_E \cup \underline{e}^*(i)$ y $\underline{e}_x \cap \underline{e}^*(i) \neq \varnothing$, entonces $e_x \in \underline{e}^*(j,\underline{e}_Q) \subseteq \underline{e}_Q \cup \underline{e}^*(j,\underline{e}_Q)$.

Por lo tanto, $e_x \in \underline{e}_Q \cup \underline{e}^*(j,\underline{e}_Q)$, es decir, $\underline{e}^*(i,j) \subseteq \underline{e}_Q \cup \underline{e}^*(j,\underline{e}_Q)$.

Sea $e_x \in \underline{e}_Q \cup \underline{e}^*(j,\underline{e}_Q)$, entonces, como $\underline{e}_Q \cap \underline{e}^*(j,\underline{e}_Q) = \varnothing$, $e_x \in \underline{e}_Q$ ó $e_x \in \underline{e}^*(j,\underline{e}_Q)$. Además, por construcción, $\underline{e}_E \cap \underline{e}^*(i) = \underline{e}^*(i) \cap \underline{e}^*(j) = \underline{e}_E \cap \underline{e}^*(j) = \varnothing$ por lo que tenemos que $e_x \in \underline{e}_E$, $e_x \in \underline{e}^*(i)$, $e_x \in \underline{e}^*(j)$ ó $e_x \in \underline{e}^*(j,\underline{e}_Q)$.

1) Si $e_x \in \underline{e}_E$, entonces $e_x = e_E \in \underline{e}_E$ y por lo tanto, $e_x = e^*(j,\underline{e}_x)$ donde $\underline{e}_x = \{e_E\}$. Es decir, $e_x \in \underline{e}^*(i,j)$.
2) Si $e_x \in \underline{e}^*(i)$, entonces $e_x = e^*(i) \in \underline{e}^*(i)$ y por lo tanto, $e_x = e^*(j,\underline{e}_x)$ donde $\underline{e}_x = \{e^*(i)\}$. Es decir, $e_x \in \underline{e}^*(i,j)$.
3) Si $e_x \in \underline{e}^*(j)$, entonces $e_x = e^*(j) = e^*(j,\underline{e}_x)$ con $\underline{e}_x \subseteq \underline{e}_E$. Es decir, $e_x \in \underline{e}^*(i,j)$.
4) Si $e_x \in \underline{e}^*(j,\underline{e}_Q)$, entonces $e_x = e^*(j,\underline{e}_x)$ con $\underline{e}_x \subseteq \underline{e}^*(i) \cup \underline{e}_E$. Por lo tanto, $e_x \in \underline{e}^*(i,j)$.

Por lo tanto, $e_x \in \underline{e}^*(i,j)$, es decir, $\underline{e}^*(i,j) \supseteq \underline{e}_Q \cup \underline{e}^*(j,\underline{e}_Q)$.

Análogamente se demuestra que $\underline{e}^*(j,i) = \underline{e}_Q \cup \underline{e}^*(i,\underline{e}_Q)$. ■

Tenemos ahora el siguiente teorema:

Teorema 5.1:

$\underline{e}^{*}(i,j) = \underline{e}^{*}(j,i)$ para cualesquiera campos i, j.

Demostración

Supondremos por simplicidad que sólo tenemos tres campos i, j y k y que comenzamos con el conjunto original de *edits* explícitos $\underline{e}_E$.

Sea $e_x \in \underline{e}_*(j, \underline{e}_Q)$. Los *edits* contribuyentes para generar a e_x están contenidos en $\underline{e}^{*}(i) \cup \underline{e}_E$. Denotemos a los subconjuntos de $\underline{e}^{*}(i)$ y $\underline{e}_E$ por $\{e_1^i, e_2^i, ..., e_{r_i}^i\}$ y $\{e_1, e_2, ..., e_r\}$, respectivamente. Ahora, como cada $e_s^i \in \{e_1^i, e_2^i, ..., e_{r_i}^i\}$ fue generado por un subconjunto de $\underline{e}_E$ denotado por $\{e_{s_1}^i, e_{s_2}^i, ..., e_{s_{n_s}}^i\}$, entonces e_x puede ser escrito como

$$e_x : \left\{ T \cap \left[A_i^{l_1} \cup A_i^{l_2} \cup \cdots \cup A_i^{l_{n_1}} \right] \cap \cdots \cap \left[A_i^{r_{i1}} \cup A_i^{r_{i2}} \cup \cdots \cup A_i^{r_{in_i}} \right] \right\} \times$$
$$\left\{ U \cup \left[A_j^{l_1} \cap A_j^{l_2} \cap \cdots \cap A_j^{l_{n_1}} \right] \cup \cdots \cup \left[A_j^{r_{i1}} \cap A_j^{r_{i2}} \cap \cdots \cap A_j^{r_{in_i}} \right] \right\} \times V = F$$

donde

$$T = A_i^1 \cap A_i^2 \cap \cdots \cap A_i^r,\ U = A_j^1 \cup A_j^2 \cup \cdots \cup A_j^r \text{ y}$$
$$V = \left[A_k^1 \cap A_k^2 \cap \cdots \cap A_k^r \right] \cap \left[A_k^{l_1} \cap A_k^{l_2} \cap \cdots \cap A_k^{l_{n_1}} \right] \cap \cdots \cap \left[A_k^{r_{i1}} \cap A_k^{r_{i2}} \cap \cdots \cap A_k^{r_{in_i}} \right].$$

Usando propiedades de conjuntos podemos escribir a e_x como

$$e_x : \left\{ \left[T \cap A_i^{l_1} \cap \cdots \cap A_i^{r_{i1}} \right] \cup \left[T \cap A_i^{l_2} \cap \cdots \cap A_i^{r_{i2}} \right] \cup \cdots \cup \left[T \cap A_i^{l_{n_1}} \cap \cdots \cap A_i^{r_{in_i}} \right] \right\} \times$$
$$\left\{ \left[U \cup A_j^{l_1} \cup \cdots \cup A_j^{r_{i1}} \right] \cap \left[U \cup A_j^{l_2} \cup \cdots \cup A_j^{r_{i2}} \right] \cap \cdots \cap \left[U \cup A_j^{l_{n_1}} \cup \cdots \cup A_j^{r_{in_i}} \right] \right\} \times V = F.$$

Entonces e_x pudo haber sido generado usando los mismos *edits* para generar e_x primero generando bajo el campo j y luego bajo el i.

Observamos que algunos de los términos entre corchetes en e_x podrían ser nulos, sin embargo, no todos los términos correspondientes al campo i ó j

pueden ser nulos ya que e_x es un *edit* válido. Entonces $\underline{e}^*(j,\underline{e}_Q) = \underline{e}^*(i,\underline{e}_Q)$. Finalmente, por el Lema 5.1., $\underline{e}^*(i,j) = \underline{e}^*(j,i)$. ■

Definamos $N_{ij} = \underline{e}^*(i,j)$. Entonces, el siguiente corolario es una consecuencia inmediata del Teorema 5.1.

Corolario 5.1:

$N_{\delta(i,j,\ldots,k)} = N_{\gamma(i,j,\ldots,k)}$ para cualesquiera dos permutaciones δ y γ de $(i,j,\ldots,k)$.

Demostración

Dadas dos permutaciones δ y γ de $(1,j,\ldots,k)$, aplicando sucesivamente el Teorema 5.1, se llega al resultado deseado. ■

Para restringir el procedimiento de generación de *edits* a una sola permutación de los campos contribuyentes, sólo requerimos el siguiente lema.

Lema 5.2:

$N_{\delta(1,2,\ldots,n)}$ es suficiente para obtener $\underline{e}_C$ para cualquier permutación δ de $(1,2,\ldots,n)$.

Demostración

Para la obtención del conjunto completo de *edits* ($\underline{e}_C$) consistente del conjunto de *edits* explícitos y del conjunto de *edits* esencialmente nuevos no redundantes es necesario realizar todos los posibles subconjuntos tanto de *edits* explícitos como implícitos que se vayan generando como *edits* contribuyentes y todos los campos como campo generador.

Por lo tanto, por el Corolario 5.1, basta con una permutación para obtener el conjunto completo de *edits*. ■

5.2 EL BOSQUE DE CÓDIGOS DE CAMPOS TIPO 1.

Supongamos que tenemos n campos, entonces el bosque de códigos de campos tipo 1 (BCC1) se construye mediante este algoritmo:

Algoritmo BCC1:

1. Por cada campo se genera el árbol i, con $i = 1,...,n$, que comienza en el nodo (i).

2. Del nodo (i) surgen $n-i$ ramas con nodos (ij), para $j = i+1,...,n$ para cada árbol i, con $i = 1,...,n$.

3. Del nodo $(ij...k)$ surgen $n-k$ ramas con nodos $(ij...km)$, $m = k+1,...,n$.

4. Se repite el paso 3 hasta que $k = n$ para todos los nodos.

El BCC1 se recorre tomando el primer árbol sin poder pasar al siguiente hasta haberlo recorrido completamente. Un árbol se recorre de arriba hacia abajo y de izquierda a derecha no pudiendo pasar a un nodo superior sin haber recorridos antes todas sus ramas.

Ejemplo 5.1:

Supongamos que tenemos un registro con cuatro campos, entonces el BCC1 será el presentado en la Figura 5.1.

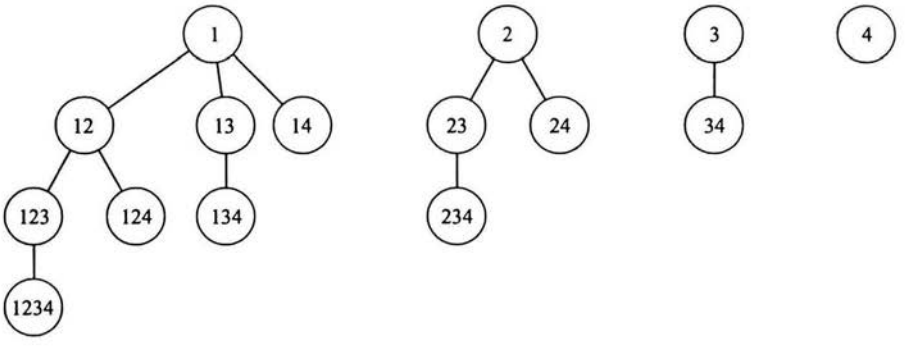


Figura 5.1. *BCC1 para el caso de cuatro campos.*

El algoritmo para obtenerlo es:

1. Generamos cuatro árboles con nodos (1), (2), (3), (4).

2. En este paso obtenemos los siguiente nodos:

Nodo raíz	Nuevos nodos
(1)	(12) (13) (14)
(2)	(23) (24)
(3)	(34)
(4)	No surgen nuevos nodos pues $k = 4$

3. En este paso obtenemos los siguientes nodos:

Nodo raíz	Nuevos nodos
(12)	(123) (124)
(13)	(134)
(14)	No surgen nuevos nodos pues $k = 4$
(23)	(234)
(24)	No surgen nuevos nodos pues $k = 4$
(34)	No surgen nuevos nodos pues $k = 4$

4. Como aún nos queda el nodo (123) repetimos el paso 3.

3'. Obtenemos como rama del (123) el nodo (1234).

4'. Como todos los nodos terminan en 4 hemos terminado el BCC1.

Observemos que el BCC1 tiene el recorrido: (1), (12), (123), (1234), (124), (13), (134), (14), (2), (23), (234), (24), (3), (34), (4).

5.3 ALGORITMO GKL1 PARA LA GENERACIÓN DEL CONJUNTO COMPLETO DE *EDITS*.

El siguiente algoritmo permite obtener el conjunto de los *edits* implícitos esencialmente nuevos no redundantes:

Algoritmo GKL1[*]*:*

1. Construya el BCC1 para el número de campos.

2. Sea $\underline{e}_G = \underline{e}_E$.

[*] Con base en el primer algoritmo dado en Robert S. Garfinkel *et al.* (1986).

3. Considere el primer elemento en el recorrido del BCC1, a saber, el nodo (1). De $\underline{e}_G$ seleccione aquellos *edits* en los que entra el campo 1. Denote a este subconjunto $\underline{e}_1$. Si $\underline{e}_1$ consiste de un solo elemento o es nulo deseche ese nodo y sus posibles ramas y retome este paso en el nodo (2). En caso contrario vaya al paso 4.

4. Considere todas las posibles parejas de *edits* en $\underline{e}_1$ y, tomando como generador al campo 1, construya un *edit* implícito por cada pareja siguiendo el procedimiento dado por el Lema 2.1. De estos nuevos *edits*, conserve únicamente los esencialmente nuevos. Si alguno de estos *edits* domina a uno en $\underline{e}_G$, lo sustituye y, si es dominado por otro en $\underline{e}_G$, se desecha. Entonces los *edits* restantes serán esencialmente nuevos no redundantes. Finalmente, agregue a $\underline{e}_G$ estos *edits*. En el caso en que no se obtenga ningún *edit* esencialmente nuevo deseche todas las posibles ramas que surjan de este nodo y retome el paso 3 en el nodo (2).

5. Pase al siguiente elemento del recorrido, a saber, el nodo (12). Entonces tome de $\underline{e}_G$ el subconjunto $\underline{e}_1$ de *edits* en los que el campo 2 entra pero no así el campo 1. Repita el paso 4 tomando como campo generador el campo 2. Si $\underline{e}_1$ consiste de un solo elemento o es nulo deseche ese nodo y sus ramas y retome este paso en el nodo (13).

6. Repita los pasos 3 y 4 para cada elemento del recorrido del BCC1, observando que, si tenemos el nodo $(i...jk)$, se considerará como $\underline{e}_1$ al subconjunto de *edits* de $\underline{e}_G$ para los cuales el campo k entra pero no así los campos $i,...,j$. Además, si $\underline{e}_1$ contiene un elemento o es nulo se desecharán sus posibles ramas del recorrido y se continuará con el siguiente nodo; de lo contrario, para cada par de *edits* en $\underline{e}_1$, se considerará como campo generador al campo k en el Lema 2.1.

7. Finalmente el conjunto $\underline{e}_G$ consistirá de los *edits* explícitos y de todos los *edits* implícitos esencialmente nuevos no redundantes.

Ejemplo 5.2:

Retomemos el Ejemplo 4.1.

Recordemos que los *edits* explícitos son:

$$e_1 : \langle A_1, \{0,1\}, \{0\}, A_4, \{0,1\}, A_6 \rangle = F,$$
$$e_2 : \langle \{1\}, A_2, \{1\}, \{0,1\}, A_5, \{2,3\} \rangle = F,$$
$$e_3 : \langle \{0\}, \{1,2\}, A_3, \{1,2,3\}, A_5, A_6 \rangle = F,$$
$$e_4 : \langle A_1, \{0,2\}, A_3, A_4, A_5, \{0,1\} \rangle = F,$$
$$e_5 : \langle \{1\}, A_2, A_3, \{0\}, \{1,2\}, A_6 \rangle = F.$$

Es decir $\underline{e}_E = \{e_1, e_2, e_3, e_4, e_5\}$.

Entonces el algoritmo es el siguiente:

1. El BCC1 correspondiente se presenta en la Figura 5.2. Debido a la gran cantidad de nodos presentes en este BCC1, ilustraremos la aplicación del algoritmo al primer árbol y al final daremos la tabla con todos los *edits* implícitos esencialmente nuevos no redundantes que se obtuvieron.

2. Tenemos que $\underline{e}_G = \{e_1, e_2, e_3, e_4, e_5\}$.

3. Consideramos el nodo (1) y observamos que $\underline{e}_I = \{e_2, e_3, e_5\}$. Como $\underline{e}_I$ tiene 3 elementos, vamos al paso 4.

4. Tenemos tres posibles parejas de *edits* en $\underline{e}_I$. Entonces, tomando como campo generador al campo 1, tenemos por el Lema 2.1 que:

$$e^*(1,\{e_2,e_3\}):\langle A_1,\{1,2\},\{1\},\{1\},A_5,\{2,3\}\rangle = F$$
$$e^*(1,\{e_2,e_5\}):\langle\{1\},A_2,\{1\},\{0\},\{1,2\},\{2,3\}\rangle = F$$
$$e^*(1,\{e_3,e_5\}):\langle A_1,\{1,2\},A_3,\varnothing,\{1,2\},A_6\rangle = F$$

   De los cuales, el único esencialmente nuevo es $e^*(1,\{e_2,e_3\})$ al no ser vacío en ninguno de sus campos y tener el campo 1 completo. Además, al no dominar ni ser dominado por ningún *edit* en $\underline{e}_G$, es un *edit* implícito esencialmente nuevo no redundante que corresponde en la Tabla 4.1 a e_6. Entonces $\underline{e}_G = \{e_1, e_2, e_3, e_4, e_5, e_6\}$.

5. Consideramos el nodo (12), entonces $\underline{e}_I = \{e_1, e_4, e_6\}$, pues en estos *edits* entra el campo 2, pero no el campo 1.

   Tenemos 3 posibles parejas de *edits*. En el Lema 2.1 tomamos el campo 2 como generador y tenemos:

$$e^*(2,\{e_1,e_4\}):\langle A_1,A_2,\{0\},A_4,\{0,1\},\{0,1\}\rangle = F$$
$$e^*(2,\{e_1,e_6\}):\langle A_1,A_2,\varnothing,\{1\},\{0,1\},\{2,3\}\rangle = F$$
$$e^*(2,\{e_4,e_6\}):\langle A_1,A_2,\{1\},\{1\},A_5,\varnothing\rangle = F$$

   De éstos, $e^*(2,\{e_1,e_4\})$ es esencialmente nuevo no redundante y corresponde en la Tabla 4.1 a e_8. Entonces $\underline{e}_G = \{e_1, e_2, e_3, e_4, e_5, e_6, e_8\}$.

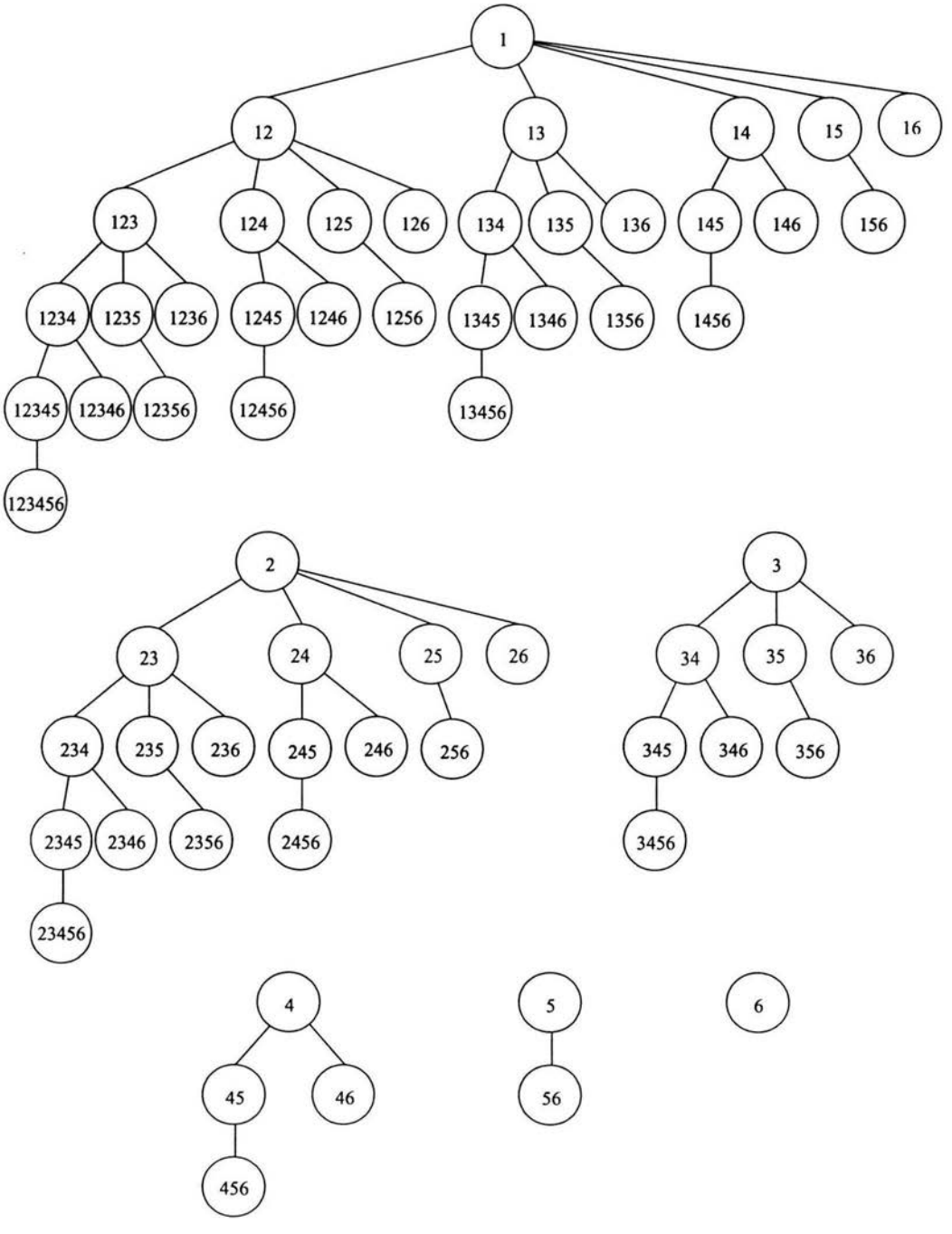


Figura 5.2. *BCC1 correspondiente al Ejemplo 5.2.*

6. El siguiente nodo en el recorrido es (123). Entonces $\underline{e}_I = \{e_8\}$, pues en este *edit* entra el campo 3, pero no los campos 2 y 1. Y como $\underline{e}_I$ tiene un solo elemento no se generan nuevos *edits*. Por lo tanto, se desechan todas sus ramas.

6.1 El siguiente nodo es (124). Entonces $\underline{e}_I = \varnothing$, por lo que se desechan todas sus ramas.

6.2 El siguiente nodo es (125). Entonces $\underline{e}_I = \{e_8\}$. Por lo tanto, no se generan nuevos *edits* y se desecha su rama.

6.3 El siguiente nodo es (126). Entonces $\underline{e}_I = \{e_8\}$. Por lo tanto, no se generan nuevos *edits*.

6.3 El siguiente nodo es (13). Entonces $\underline{e}_I = \{e_1, e_6, e_8\}$.

Tenemos 3 posibles parejas de *edits*. En el Lema 2.1 tomamos el campo 3 como generador y tenemos:

$$e^*(3,\{e_1,e_6\}):\langle A_1,\{1\},A_3,\{1\},\{0,1\},\{2,3\}\rangle = F$$
$$e^*(3,\{e_1,e_8\}):\langle A_1,\{0,1\},\{0\},A_4,\{0,1\},\{0,1\}\rangle = F$$
$$e^*(3,\{e_6,e_8\}):\langle A_1,\{1,2\},A_3,\{1\},\{0,1\},\varnothing\rangle = F$$

De éstos, $e^*(3,\{e_1,e_6\})$ es esencialmente nuevo no redundante y corresponde en la Tabla 4.1 a e_{11}. Entonces $\underline{e}_G = \{e_1,e_2,e_3,e_4,e_5,e_6,e_8,e_{11}\}$.

6.5 El siguiente nodo es (134). Entonces $\underline{e}_I = \{e_{11}\}$. Por lo tanto, no se generan nuevos *edits* y se desechan todas sus ramas.

6.6 El siguiente nodo es (135). Entonces $\underline{e}_I = \{e_{11}\}$. Por lo tanto, no se generan nuevos *edits* y se desecha su rama.

6.7 El siguiente nodo es (136). Entonces $\underline{e}_I = \{e_4, e_{11}\}$. Tenemos una posible pareja de *edits*. En el Lema 2.1 tomamos el campo 6 como generador y tenemos:

$$e^*(6,\{e_4,e_{11}\}):\langle A_1,\varnothing,A_3,\{1\},\{0,1\},A_6\rangle = F$$

Pero éste no es un *edit* válido, por lo que no se modifica $\underline{e}_G$.

6.8 El siguiente nodo es (14). Entonces $\underline{e}_I = \{e_6, e_{11}\}$. Tenemos una posible pareja de *edits*. En el Lema 2.1 tomamos el campo 4 como generador y tenemos:

$$e^*(4,\{e_6,e_{11}\}):\langle A_1,\{1\},\{1\},\{1\},\{0,1\},\{2,3\}\rangle = F$$

Pero este *edit* no es esencialmente nuevo, por lo que desechamos todas las posibles ramas que surgen de este nodo.

6.9 El siguiente nodo es (15). Entonces $\underline{e}_I = \{e_1, e_8, e_{11}\}$. Tenemos 3 posibles parejas de *edits*. En el Lema 2.1 tomamos el campo 5 como generador y tenemos:

$$e^*(5,\{e_1,e_8\}):\langle A_1,\{0,1\},\{0\},A_4,\{0,1\},\{0,1\}\rangle = F$$
$$e^*(5,\{e_1,e_{11}\}):\langle A_1,\{1\},\{0\},\{1\},\{0,1\},\{2,3\}\rangle = F$$
$$e^*(5,\{e_8,e_{11}\}):\langle A_1,\{1\},\{0\},\{1\},\{0,1\},\varnothing\rangle = F$$

Pero ninguno de estos *edits* es esencialmente nuevo, por lo que desechamos la rama que surge de este nodo.

6.10 El siguiente nodo es (16). Entonces $\underline{e}_I = \{e_4, e_6, e_8, e_{11}\}$. Tenemos 6 posibles parejas de *edits*. En el Lema 2.1 tomamos el campo 6 como generador y tenemos:

$$e^*(6,\{e_4,e_6\}):\langle A_1,\{2\},\{1\},\{1\},A_5,A_6\rangle = F$$
$$e^*(6,\{e_4,e_8\}):\langle A_1,\{0,2\},\{0\},A_4,\{0,1\},\{0,1\}\rangle = F$$
$$e^*(6,\{e_4,e_{11}\}):\langle A_1,\varnothing,A_3,\{1\},\{0,1\},A_6\rangle = F$$
$$e^*(6,\{e_6,e_8\}):\langle A_1,\{1,2\},\varnothing,\{1\},\{0,1\},A_6\rangle = F$$
$$e^*(6,\{e_6,e_{11}\}):\langle A_1,\{1\},\{1\},\{1\},\{0,1\},\{2,3\}\rangle = F$$
$$e^*(6,\{e_8,e_{11}\}):\langle A_1,\{1\},\{0\},\{1\},\{0,1\},A_6\rangle = F$$

De éstos, $e^*(6,\{e_4,e_6\})$ es esencialmente nuevo no redundante y corresponde en la Tabla 4.1 a e_{16}, mientras que $e^*(6,\{e_8,e_{11}\})$ se desecha porque es dominado por e_1. Entonces $\underline{e}_G = \{e_1,e_2,e_3,e_4,e_5,e_6,e_8,e_{11},e_{16}\}$.

El algoritmo se sigue con los demás árboles del BCC1.

7. La siguiente tabla nos muestra los *edits* implícitos esencialmente nuevos no redundantes, sus *edits* contribuyentes y el nodo en el que se obtuvieron.

Nodo	*Edits* Contribuyentes	*Edit* implícito esencialmente nuevo no redundante
(1)	e_2, e_3	$e_6 : \langle A_1, \{1,2\}, \{1\}, \{1\}, A_5, \{2,3\} \rangle = F$
(12)	e_1, e_4	$e_8 : \langle A_1, A_2, \{0\}, A_4, \{0,1\}, \{0,1\} \rangle = F$
(13)	e_1, e_6	$e_{11} : \langle A_1, \{1\}, A_3, \{1\}, \{0,1\}, \{2,3\} \rangle = F$
(16)	e_4, e_6	$e_{16} : \langle A_1, \{2\}, \{1\}, \{1\}, A_5, A_6 \rangle = F$
(2)	e_1, e_3	$e_7 : \langle \{0\}, A_2, \{0\}, \{1,2,3\}, \{0,1\}, A_6 \rangle = F$
	e_3, e_4	$e_9 : \langle \{0\}, A_2, A_3, \{1,2,3\}, A_5, \{0,1\} \rangle = F$
(25)	e_5, e_8	$e_{13} : \langle \{1\}, A_2, \{0\}, \{0\}, A_5, \{0,1\} \rangle = F$
(3)	e_1, e_2	$e_{10} : \langle \{1\}, \{0,1\}, A_3, \{0,1\}, \{0,1\}, \{2,3\} \rangle = F$
(35)	e_5, e_{10}	$e_{14} : \langle \{1\}, \{0,1\}, A_3, \{0\}, A_5, \{2,3\} \rangle = F$
(356)	e_4, e_{14}	$e_{18} : \langle \{1\}, \{0\}, A_3, \{0\}, A_5, A_6 \rangle = F$
(36)	e_4, e_{10}	$e_{17} : \langle \{1\}, \{0\}, A_3, \{0,1\}, \{0,1\}, A_6 \rangle = F$
(5)	e_1, e_5	$e_{12} : \langle \{1\}, \{0,1\}, \{0\}, \{0\}, A_5, A_6 \rangle = F$
(56)	e_2, e_4	$e_{15} : \langle \{1\}, \{0,2\}, \{1\}, \{0,1\}, A_5, A_6 \rangle = F$

5.4 ALGORITMO GLK2 PARA LA IMPUTACIÓN DE DATOS EN REGISTROS PARTICULARES.

En la Sección 5.3 presentamos un algoritmo para generar el conjunto completo de *edits* $\underline{e}_C$. Si el conjunto es prohibitivamente caro de generar, entonces una opción para generar sólo aquellos *edits* implícitos asociados con un registro específico $y^0 \in \underline{E}_E$ sería deseable. El siguiente es un algoritmo diseñado para este propósito.

Algoritmo GKL2[•]*:*

1. Resuelva el PSC (4.8) y denote la cubierta solución x^*.

2. Sea $J^* = \{j \mid x_j^* = 1\}$. Fije los valores y_j^0 con $j \notin J^*$ en, pero para cada $j \in J^*$ tome y_j^0 todos los valores de A_j. Seleccione de los $\prod_{j \in J^*} n_j$ posibles registros aquellos que pertenezcan a $A - \underline{E}_E$. Si alguno de estos registros es encontrado, x^* especifica una solución para el MCPI. En caso contrario, vaya al paso 3.

3. Encuentre cualquier cubierta solución v^0 de

[•] Con base en el segundo algoritmo dado en Robert S. Garfinkel *et al.* (1986).

$$vQ_k \geq 1 \qquad (5.1)$$

donde:

v es un vector binario de tamaño el número de *edits* en $\underline{e}_F(y^0)$ y

Q_k es la k-ésima columna de Q con

$$Q=(q_{ik}) \text{ y } q_{ik}=\begin{cases}1 \text{ si } e_i \in \underline{e}_F(y^0) \text{ es fallado por el } k\text{-ésimo registro} \\ \quad \text{obtenido en el paso 2} \\ 0 \text{ en otro caso}\end{cases}$$

Sea $I^0=\{i \mid v_i^0=1\}$.

4. Genere el *edit* implícito $\hat{e}: \hat{E}=F$ con

$$\hat{A}_j=\begin{cases}\bigcap_{i\in I^0} A_j^i, & j \notin J^* \\ \bigcup_{i\in I^0} A_j^i, & j \in J^*\end{cases} \qquad (5.2)$$

Sea $\underline{e}_F(y^0)=\underline{e}_F(y^0)\cup\{\hat{e}\}$ y vaya al paso 1.

Ejemplo 5.3:

Consideremos los *edits* explícitos del Ejemplo 5.2, $c=(5 \;\; 3 \;\; 1 \;\; 4 \;\; 2 \;\; 4)$ y $y^0=(1 \;\; 0 \;\; 0 \;\; 0 \;\; 1 \;\; 0)$. Entonces el algoritmo es el siguiente:

1. Observemos que $\underline{e}_F(y^0)=\{e_1,e_4,e_5\}$. Entonces el PSC (4.8) es:

$$\begin{array}{lrcrcrcrcrcl} \text{Minimizar} & 5x_1 & + & 3x_2 & + & x_3 & + & 4x_4 & + & 2x_5 & + 4x_6 & \\ \text{Sujeto a} & & & x_2 & + & x_3 & + & & & x_5 & & \geq 1 \\ & & & x_2 & + & & & & & & x_6 & \geq 1 \\ & x_1 & + & & & & & x_4 & + & x_5 & & \geq 1 \\ & & & & & x_1, \ldots, x_6 = 0, 1 & & & & & & \end{array}$$

La cubierta solución es $x^*=(0 \;\; 1 \;\; 0 \;\; 0 \;\; 1 \;\; 0)$ pues x_2 y x_5 juntas cubren todas las restricciones. El costo es 5.

2. Tenemos que $J^*=\{2, 5\}$. Como $n_2=n_5=3$, tenemos 9 posibles registros a verificar dados en la siguiente tabla junto con los *edits* que fallan.

Posible registro	***Edit(s)* fallado(s) por el registro**
(1 0 0 0 0 0)	e_1, e_4
(1 1 0 0 0 0)	e_1
(1 2 0 0 0 0)	e_4
(1 0 0 0 1 0)	e_1, e_4, e_5
(1 1 0 0 1 0)	e_1, e_5
(1 2 0 0 1 0)	e_4, e_5
(1 0 0 0 2 0)	e_4, e_5
(1 1 0 0 2 0)	e_5
(1 2 0 0 2 0)	e_4, e_5

Observamos que los valores y_j^0 con $j \notin J^*$ están fijos. Como ningún registro satisface todos los *edits*, es decir, pertenece a $A - \underline{E}_E$, vamos al paso 3.

3. Tenemos que

$$Q = \begin{pmatrix} 1 & 1 & 0 & 1 & 1 & 0 & 0 & 0 & 0 \\ 1 & 0 & 1 & 1 & 0 & 1 & 1 & 0 & 1 \\ 0 & 0 & 0 & 1 & 1 & 1 & 1 & 1 & 1 \end{pmatrix}$$

Recordemos que las columnas corresponden a los 9 posibles registros y las filas a los 3 *edits* fallados por el registro original.

Entonces una cubierta principal solución para (5.1) es $v^0 = (1 \quad 1 \quad 1)$ pues $v^0 Q_1 = 2$, $v^0 Q_2 = 1$, $v^0 Q_3 = 1$, $v^0 Q_4 = 3$, $v^0 Q_5 = 2$, $v^0 Q_6 = 2$, $v^0 Q_7 = 2$, $v^0 Q_8 = 1$ y $v^0 Q_9 = 2$. Entonces la cubierta principal será $I^0 = \{1, 4, 5\}$.

4. El *edit* implícito generado es:

$$\hat{e} : \langle \{1\}, A_2, \{0\}, \{0\}, A_5, \{0,1\} \rangle = F$$

que corresponde a e_{13} en la Tabla 4.1. Entonces $\underline{e}_F(y^0) = \{e_1, e_4, e_5, e_{13}\}$.

1'. Al PSC (4.8) del paso 1 agregamos la restricción $x_1 + x_3 + x_4 + x_6 \geq 1$ dada por e_{13}. Entonces una cubierta solución es $x^* = (0 \quad 0 \quad 0 \quad 0 \quad 1 \quad 1)$ con costo 6.

2'. Tenemos que $J^* = \{5, 6\}$. Como $n_5 = 3$ y $n_6 = 4$, tenemos 12 posibles registros a verificar dados en la siguiente tabla junto con los *edits* que fallan.

Posible registro	***Edit(s)* fallado(s) por el registro**
(1 0 0 0 0 0)	e_1, e_4, e_{13}
(1 0 0 0 0 1)	e_1, e_4, e_{13}
(1 0 0 0 0 2)	e_1
(1 0 0 0 0 3)	e_1
(1 0 0 0 1 0)	e_1, e_4, e_5, e_{13}
(1 0 0 0 1 1)	e_1, e_4, e_5, e_{13}
(1 0 0 0 1 2)	e_1, e_5
(1 0 0 0 1 3)	e_1, e_5
(1 0 0 0 2 0)	e_4, e_5, e_{13}
(1 0 0 0 2 1)	e_4, e_5, e_{13}
(1 0 0 0 2 2)	e_5
(1 0 0 0 2 3)	e_5

Como ningún registro satisface todos los *edits*, es decir, pertenece a $A - \underline{E}_E$, pasamos al paso 3.

3'. Tenemos que

$$Q = \begin{pmatrix} 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 & 1 & 1 & 0 & 0 & 1 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 0 & 0 & 1 & 1 & 0 & 0 & 1 & 1 & 0 & 0 \end{pmatrix}$$

Entonces una cubierta principal solución para (5.1) es $v^0 = (1 \;\; 0 \;\; 1 \;\; 0)$ pues $v^0 Q_1 = 1$, $v^0 Q_2 = 1$, $v^0 Q_3 = 1$, $v^0 Q_4 = 1$, $v^0 Q_5 = 2$, $v^0 Q_6 = 2$, $v^0 Q_7 = 2$, $v^0 Q_8 = 2$, $v^0 Q_9 = 1$, $v^0 Q_{10} = 1$, $v^0 Q_{11} = 1$ y $v^0 Q_{12} = 1$. Así la cubierta principal es $I^0 = \{1, 5\}$.

4'. El *edit* implícito generado es:

$$\hat{e} : \langle \{1\}, \{0,1\}, \{0\}, \{0\}, A_5, A_6 \rangle = F$$

que corresponde a e_{12} en la Tabla 4.1. Entonces, $\underline{e}_F(y^0) = \{e_1, e_4, e_5, e_{12}, e_{13}\}$.

1''. Al PSC (4.8) del paso 1' agregamos la restricción $x_1 + x_2 + x_3 + x_4 + \geq 1$ dada por e_{12}. Entonces, una solución es $x^* = (0 \quad 1 \quad 1 \quad 0 \quad 1 \quad 0)$ con costo 6.

2''. Tenemos que $J^* = \{2, 3, 5\}$. Como $n_2 = 3$, $n_3 = 2$ y $n_5 = 3$, tenemos 18 posibles registros a verificar dados en la siguiente tabla junto con los *edits* que fallan.

Posible registro	***Edit*(s) fallado(s) por el registro**
$(1 \quad 0 \quad 0 \quad 0 \quad 0 \quad 0)$	e_1, e_4, e_{12}, e_{13}
$(1 \quad 0 \quad 0 \quad 0 \quad 1 \quad 0)$	$e_1, e_4, e_5, e_{12}, e_{13}$
$(1 \quad 0 \quad 0 \quad 0 \quad 2 \quad 0)$	e_4, e_5, e_{12}, e_{13}
$(1 \quad 0 \quad 1 \quad 0 \quad 0 \quad 0)$	e_4, e_{12}
$(1 \quad 0 \quad 1 \quad 0 \quad 1 \quad 0)$	e_4, e_5, e_{12}
$(1 \quad 0 \quad 1 \quad 0 \quad 2 \quad 0)$	e_4, e_5, e_{12}
$(1 \quad 1 \quad 0 \quad 0 \quad 0 \quad 0)$	e_1, e_{13}
$(1 \quad 1 \quad 0 \quad 0 \quad 1 \quad 0)$	e_1, e_5, e_{13}
$(1 \quad 1 \quad 0 \quad 0 \quad 2 \quad 0)$	e_5, e_{13}
$(1 \quad 1 \quad 1 \quad 0 \quad 0 \quad 0)$	
$(1 \quad 1 \quad 1 \quad 0 \quad 1 \quad 0)$	e_5
$(1 \quad 1 \quad 1 \quad 0 \quad 2 \quad 0)$	e_5
$(1 \quad 2 \quad 0 \quad 0 \quad 0 \quad 0)$	e_4, e_{13}
$(1 \quad 2 \quad 0 \quad 0 \quad 1 \quad 0)$	e_4, e_5, e_{13}
$(1 \quad 2 \quad 0 \quad 0 \quad 2 \quad 0)$	e_4, e_5, e_{13}
$(1 \quad 2 \quad 1 \quad 0 \quad 0 \quad 0)$	e_4
$(1 \quad 2 \quad 1 \quad 0 \quad 1 \quad 0)$	e_4, e_5
$(1 \quad 2 \quad 1 \quad 0 \quad 2 \quad 0)$	e_4, e_5

De aquí observamos que el registro $(1 \quad 1 \quad 1 \quad 0 \quad 0 \quad 0)$ está en $A - \underline{E}_E$, es decir, no falla ningún *edit*.

Notamos que cuando el algoritmo termina en el paso 2'', da una imputación óptima en vez de simplemente los campos óptimos a imputar. Por otro lado, si hay imputaciones óptimas alternativas, tendría sentido utilizar criterios adicionales para escoger cuál imputación usar.

En este ejemplo es interesante notar que sólo 2 de los 13 *edits* implícitos esencialmente nuevos no redundantes de la Tabla 4.1 fueron necesarios. También observamos que la mitad de los campos van a ser cambiados. En tal situación, el registro no aporta información y descartarlo podría ser la mejor alternativa. De esta manera podría ser razonable agregar al paso 1 la condición de que, si el costo de x^* excede una cota dada, el registro será descartado. Tal opción impediría al paso 2 volverse demasiado extenso al limitar el crecimiento de $\prod_{j \in J^*} n_j$.

Encontrar una cubierta principal para (5.1) en el paso 3 es mucho más fácil que resolver un PSC. Un procedimiento simple es primero encontrar cualquier cubierta y después descartar de manera secuencial cualquier *edit* que no sea responsable único de cubrir al menos un registro. También no es difícil mostrar que la construcción del paso 4 en realidad genera un *edit* implícito esencialmente nuevo, el cuál es fallado por el registro y^0.

Note que (5.2) generaliza la construcción dada por el Lema 2.1 para la generación en más de un campo, pero sólo para la situación en la que el paso 2 ha sido ya realizado. En general, no es difícil mostrar que la generación sobre más de un campo puede no producir un *edit* implícito valido.

5.5 ALGUNAS CONSIDERACIONES SOBRE LOS DOS ALGORITMOS.

El primer algoritmo requiere la generación del conjunto completo, el cual puede ser bastante grande. Y esto sin considerar que aún nos faltaría realizar la imputación para la cual se requeriría aplicar los algoritmos del Capítulo III.

El segundo algoritmo resuelve una secuencia de PSC's para cada registro fallado. El propósito es encontrar una imputación factible generando sólo una fracción pequeña del conjunto completo de forma que el PSC será relativamente pequeño. El principal problema en cuanto al tiempo de computadora se dará en el paso 2, en el que se deberá probar un gran número de imputaciones potenciales y en el número de PSC's a resolver.

Aparentemente, para problemas que pueden ser resueltos rápidamente por ambos algoritmos, si el tiempo para generar el conjunto completo de *edits* es pequeño, entonces el primer algoritmo se preferirá al segundo. En caso contrario, el segundo algoritmo se desempeñará mejor.

CAPÍTULO VI
ALGORITMO WW

En este capítulo se presenta un algoritmo de generación de *edits*, llamado el Algoritmo WW, que es alternativo al Algoritmo GKL1. Al igual que éste, el Algoritmo WW reduce los cálculos con relación a los algoritmos directamente basados en las ideas de Fellegi y Holt. Los resultados teóricos son una conjugación de los resultados en los capítulos II y V. Proporcionamos un ejemplo, basado en el Ejemplo 4.1, que muestra que el Algoritmo WW genera el conjunto de *edits* implícitos en menos pasos que el Algoritmo GKL1.

Este capítulo tiene la intención de reemplazar la generación de *edits* implícitos dada por el Algoritmo GKL1 ya que, mientras los Algoritmos GKL son mejores a nivel computacional que los dados en el Capítulo II, se visitan nodos que podrían ser eliminados y que, por ende, consumen tiempo.

El algoritmo de generación de *edits* de este capítulo conserva mucha de la superioridad computacional del Algoritmo GKL1 y es más cercano a las ideas originales de Fellegi y Holt.

6.1. RESULTADOS TEÓRICOS.

Como hemos observado antes, si un *edit* e_i redundante es parte del conjunto generador de *edits* $\underline{e}_G$ entonces el *edit* implícito generado será dominado (redundante) por el *edit*

implícito que es generado por $\underline{e}_{GM} = \underline{e}_G / \{e_i\} \cup \{e_j\}$ donde e_j es un *edit* que domina a e_i. De esta manera, si somos capaces de restringir la generación de *edits* a subconjuntos que contengan *edits* no redundantes entonces podemos reducir los cálculos.

Dado que el Teorema 2.2 demuestra que el procedimiento del Lema 2.1 genera todos los *edits* implícitos no redundantes, podemos, al realizar la generación de *edits*, llevar un registro de los *edits* generados que reemplacen a los *edits* explícitos. $\underline{e}_{EM}$ es el conjunto de los *edits* implícitos no redundantes que reemplaza a los *edits* explícitos más el conjunto de los *edits* explícitos que no son reemplazados. Si realizamos el procedimiento de generación de *edits* de nuevo con el $\underline{e}_{EM}$ como conjunto inicial de *edits*, generamos $\underline{e}_C$.

Cualquier *edit* generado que reemplace un *edit* explícito será aún llamado un *edit* explícito. Dado que $\underline{e}_{EM}$ puede generar menos *edits* redundantes, los cálculos pueden ser reducidos. El conjunto $\underline{e}_{EM}$ se dice que es ***equivalente*** a $\underline{e}_E$ porque genera el mismo conjunto de *edits* implícitos esencialmente nuevos no redundantes.

6.2. EL BOSQUE DE CÓDIGOS DE CAMPOS TIPO 2.

Para aplicar el Algoritmo WW es necesario construir primero el bosque de códigos de campos tipo 2 (BCC2) que difiere del BCC1 en que en éste tendremos nodos de la forma $(ij...k)$ con $i, j, k \in \{1,...,n\}$, mientras que en el del Capítulo V los nodos eran de la forma $(ij...k)$ con $i < j < k$.

El algoritmo para general el BCC2 es el siguiente:

Algoritmo BCC2:

1. Por cada campo genere un árbol $i, i \in \{1,...,n\}$, que comienza en el nodo (i).

2. Del nodo (i) surgen $n-1$ ramas con nodos (ij), $j \in \{1,...,n\}/\{i\}$ para cada árbol $i, i \in \{1,...,n\}$.

3. Del nodo $(ij...k)$ surgen $n-a$ ramas, donde a = número de elementos en el nodo $(ij...k)$, con nodos $(ij...km)$, $m \in \{1,...,n\}/\{i, j,...,k\}$.

4. Repita el paso 3 hasta que obtenga todos los posibles nodos con n elementos.

Ejemplo 6.1:

Supongamos que tenemos un registro con cuatro campos, entonces el BCC2 será el presentado en la Figura 6.1.

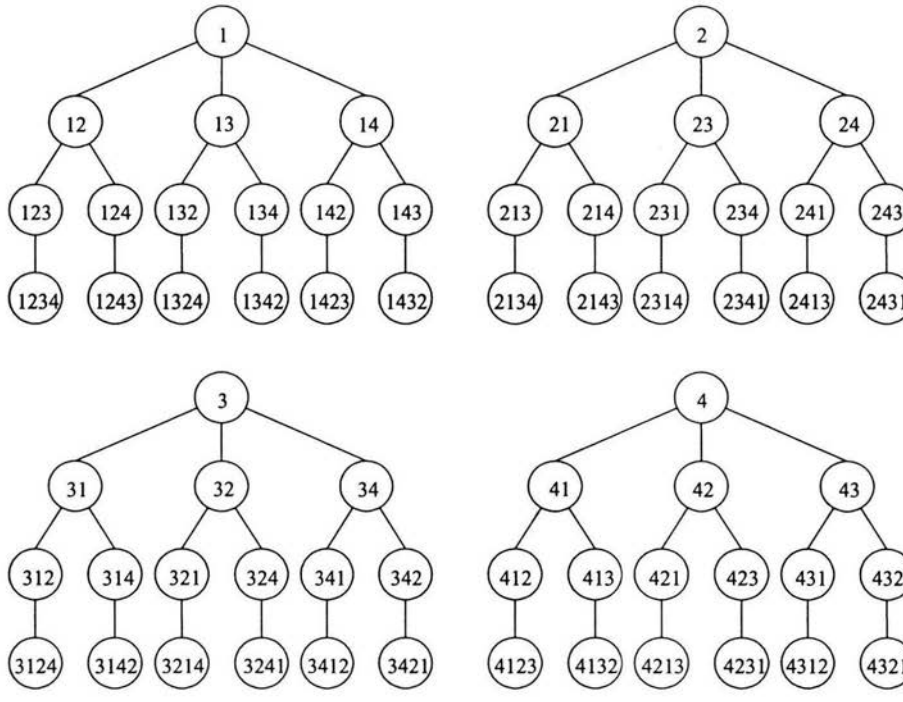


Figura 6.1. *BCC2 para el caso de cuatro campos.*

El algoritmo para obtenerlo es:

1. Generamos cuatro árboles con nodos (1), (2), (3), (4).

2. En este paso obtenemos los siguiente nodos:

Nodo raíz	**Nuevos nodos**
(1)	(12), (13), (14)
(2)	(21), (23), (24)
(3)	(31), (32), (34)
(4)	(41), (42), (43)

3. En este paso obtenemos los siguientes nodos:

Nodo raíz	Nuevos nodos
(12)	(123), (124)
(13)	(132), (134)
(14)	(142), (143)
(21)	(213), (214)
(23)	(231), (234)
(24)	(241), (243)
(31)	(312), (314)
(32)	(321), (324)
(34)	(341), (342)
(41)	(412), (413)
(42)	(421), (423)
(43)	(431), (432)

4. Como aún los nodos no tienen 4 elementos repetimos el paso 3.

3'. En este paso obtenemos los siguientes nodos:

Nodo raíz	Nuevos nodos
(123)	(1234)
(124)	(1243)
(132)	(1324)
(134)	(1342)
(142)	(1423)
(143)	(1432)
(213)	(2134)
(214)	(2143)
(234)	(2341)
(241)	(2413)
(243)	(2431)
(312)	(3124)
(314)	(3142)
(321)	(3214)
(324)	(3241)
(341)	(3412)
(342)	(3421)
(412)	(4123)
(413)	(4132)
(421)	(4213)
(423)	(4231)
(431)	(4312)
(432)	(4321)

4'. Como todos los nodos tienen 4 elementos hemos terminado el BCC2.

6.3 ALGORITMO WW PARA LA GENERACIÓN DEL CONJUNTO COMPLETO DE *EDITS*.

Una vez construido el BCC2, el siguiente algoritmo permite obtener el conjunto de los *edits* implícitos esencialmente nuevos no redundantes:

*Algoritmo WW**:

1. Reemplace, si es necesario, el conjunto original de *edits* explícitos $\underline{e}_E$ por un conjunto equivalente de *edits* explícitos no redundantes $\underline{e}_{EM}$.

2. Considere el nodo (i) del primer árbol, a saber, el nodo (1). De $\underline{e}_{EM}$ seleccione aquellos *edits* en los que entra el campo *i*. Denótese a este subconjunto $\underline{e}_I$. Si $\underline{e}_I$ consta de un solo elemento o es vacío deseche ese nodo y sus posibles ramas y vaya al paso 8. En caso contrario vaya al paso 3.

3. Considere todas las posibles parejas de *edits* en $\underline{e}_I$ y, tomando como generador al campo *i*, construya un *edit* implícito para cada pareja siguiendo el procedimiento dado por el Lema 2.1. De estos *edits* implícitos conserve únicamente los esencialmente nuevos no redundantes. En el caso en que no se obtenga ningún *edit* de este tipo deseche del recorrido todas las ramas que surgen de este nodo y vaya al paso 8. En caso contrario vaya al paso 4.

4. De los nodos (ij), conserve aquellos en los que el campo *j* entra en alguno de los *edits* implícitos esencialmente nuevos no redundantes generados en el paso anterior. Sea $C_i = \{c_{ij}\}$ el conjunto de los nodos que conservó. Sea $m_i = \#(C_i)$, es decir, el número de elementos del conjunto C_i.

5. Considere el primer elemento de C_i, a saber, c_{i1}. Seleccione de $\underline{e}_{EM}$ aquellos *edits* en los que dicho campo entra, pero no así el campo *i*. Denótese por $\underline{e}_g$ a este conjunto de *edits*. En caso de que $\underline{e}_g$ sea vacío pase al siguiente elemento de C_i y retome este paso (si no resta ningún elemento en C_i vaya al paso 8). En caso contrario vaya al paso 6.

6. Considere todas las posibles parejas de *edits* esencialmente nuevos del paso 3 con *edits* en $\underline{e}_g$. Por cada pareja genere un *edit* implícito con el campo c_{ij} como generador en el Lema 2.1. De estos *edits* conserve únicamente los esencialmente nuevos no redundantes. En el caso en que no se obtenga ningún *edit* esencialmente

* Con base en el algoritmo dado en William E. Winkler (1997).

nuevo no redundante deseche todas las posibles ramas que surgen de este nodo y retome el paso 5 para el siguiente elemento en C_i (si no resta ningún elemento vaya al paso 8). En caso contrario vaya al paso 7.

7. Repita los pasos 4, 5 y 6 para el nodo (ic_{ij}) del paso anterior, observando que ahora se generará un conjunto $C_{ij} = \{c_{ijk}\}$ con m_{ij} elementos y que en el caso en que agote los elementos de C_{ij} pasará al siguiente elemento de C_i (en caso de que no reste ningún elemento en C_i se pasará al paso 8). Este paso será iterativo hasta que se llegue al nodo $(ic_{im_i} c_{im_{m_i}} \cdots c_{im_{m_{i-1}}})$ con n elementos o se eliminen todas las ramas de un nivel en el árbol.

8. Agregue a $\underline{e}_{EM}$ los *edits* esencialmente nuevos no redundantes generados en los pasos 3 y 6. Repita el paso 2 para el siguiente nodo en el recorrido hasta completar todo el BCC2.

Ejemplo 6.2:

Este ejemplo está basado en el Ejemplo 4.1.

Consideremos los siguientes *edits* explícitos:

$$e_1 : \langle A_1, \{0,1\}, \{0\}, A_4, \{0,1\}, A_6 \rangle = F,$$
$$e_2 : \langle \{1\}, A_2, \{1\}, \{0,1\}, A_5, \{2,3\} \rangle = F,$$
$$e_3 : \langle \{0\}, \{1,2\}, A_3, \{1,2,3\}, A_5, A_6 \rangle = F,$$
$$e_4 : \langle A_1, \{0,2\}, A_3, A_4, A_5, \{0,1\} \rangle = F,$$
$$e_5 : \langle \{1\}, A_2, A_3, \{0\}, \{1,2\}, A_6 \rangle = F.$$

Es decir $\underline{e}_E = \{e_1, e_2, e_3, e_4, e_5\}$.

Debido a la extensión del BCC2, presentamos en la Figura 6.2 sólo los nodos que se utilizaron en la aplicación del Algoritmo WW.

Algoritmo WW:

1. Nos damos cuenta que $\underline{e}_E = \{e_1, e_2, e_3, e_4, e_5\}$ es un conjunto de *edits* no redundantes, pues ninguno está contenido en otro. Por lo tanto $\underline{e}_{EM} = \underline{e}_E$.

2. Consideramos el nodo (1) y observamos que $\underline{e}_1 = \{e_2, e_3, e_5\}$.

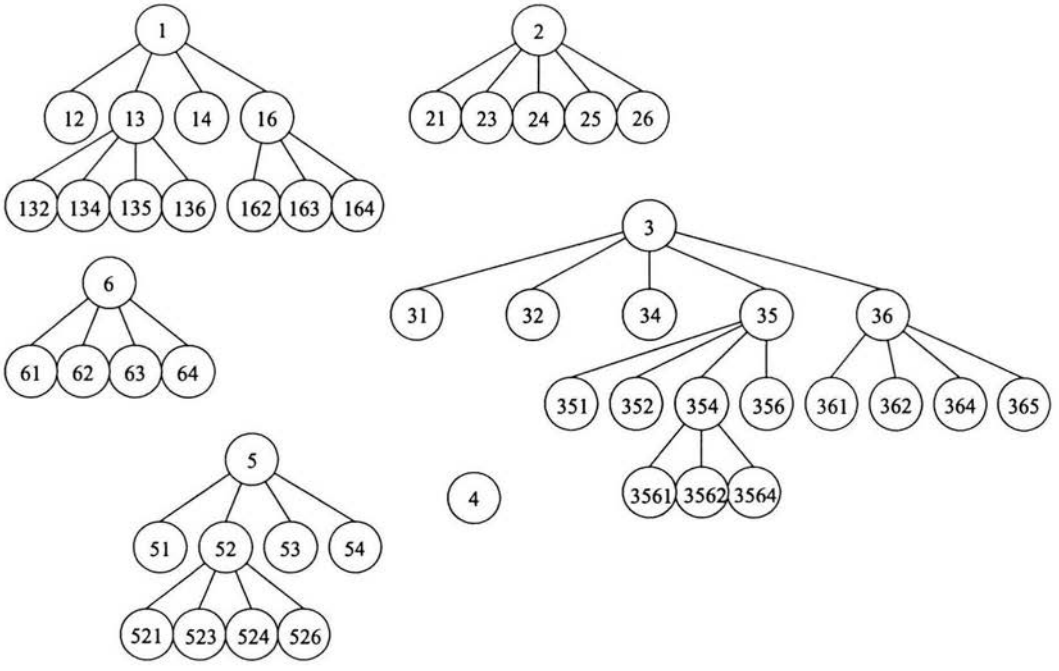


Figura 6.2. *BCC2 correspondiente al Ejemplo 6.2.*

3. Tenemos 3 posibles parejas de *edits* en $\underline{e}_1$, entonces, tomando como campo generador al campo 1, tenemos que, del Lema 2.1:

$$e^*\left(1,\{e_2,e_3\}\right):\langle A_1,\{1,2\},\{1\},\{1\},A_5,\{2,3\}\rangle = F$$
$$e^*\left(1,\{e_2,e_5\}\right):\langle\{1\},A_2,\{1\},\{0\},\{1,2\},\{2,3\}\rangle = F$$
$$e^*\left(1,\{e_3,e_5\}\right):\langle A_1,\{1,2\},A_3,\varnothing,\{1,2\},A_6\rangle = F$$

   De los cuales, el único esencialmente nuevos es $e^*\left(1,\{e_2,e_3\}\right)$ que, al no dominar ni ser dominado por ningún *edit* en $\underline{e}_E$, es un *edit* implícito esencialmente nuevo no redundante que corresponde en la Tabla 4.1 a e_6.

4. De los nodos (12), (13), (14), (15) y (16) conservamos únicamente los nodos (12), (13), (14) y (16) pues en e_6 (que fue el *edit* esencialmente nuevo no redundante generado en el nodo superior) el campo 5 no entra en él. Entonces C_1 = {2, 3, 4, 6} y $m_1 = 4$.

5. Consideramos el nodo (12). Entonces $\underline{e}_g = \{e_1, e_4\}$, pues en estos *edits* entra el campo 2, pero no el campo 1.

6. Tenemos 2 posibles parejas de *edits*: $\{e_6, e_1\}$ y $\{e_6, e_4\}$. Entonces, aplicando el Lema 2.1, tenemos:

$$e^*\left(2, \{e_6, e_1\}\right): \langle A_1, A_2, \varnothing, \{1\}, \{0,1\}, \{2,3\}\rangle = F$$
$$e^*\left(2, \{e_6, e_4\}\right): \langle A_1, A_2, \{1\}, \{1\}, A_5, \varnothing\rangle = F$$

Observamos que ninguno es válido, por lo que desechamos todas las ramas del nodo.

5.1 Consideramos el nodo (13). Entonces $\underline{e}_g = \{e_1\}$, pues en este *edit* entra el campo 3, pero no el campo 1.

6.1 Tenemos una pareja de *edits*, $\{e_6, e_1\}$, que produce el *edit* implícito $e^*\left(3, \{e_6, e_1\}\right): \langle A_1, \{1\}, A_3, \{1\}, \{0,1\}, \{2,3\}\rangle = F$, el cual es esencialmente nuevo no redundante y corresponde en la Tabla 4.1 a e_{11}.

7. De los nodos (132), (134), (135) y (136) nos quedamos con todos, pues los campos 2, 4, 5 y 6 entran en el *edit* e_{11} generado en el paso anterior. Entonces C_{13} = {2, 4, 5, 6} y $m_{13} = 4$.

Consideramos el nodo (132). Entonces $\underline{e}_g = \{e_4\}$. Tenemos la pareja $\{e_{11}, e_4\}$ que genera $e^*\left(2, \{e_{11}, e_4\}\right): \langle A_1, A_2, A_3, \{1\}, \{0,1\}, \varnothing\rangle = F$, el cual no es válido, por lo que desechamos todas sus ramas.

Consideramos ahora el nodo (134). Entonces $\underline{e}_g = \varnothing$, por lo que desechamos todas sus ramas. Análogamente, para el nodo (135), $\underline{e}_g = \varnothing$, por lo que desechamos todas sus ramas.

Consideramos el nodo (136). Entonces $\underline{e}_g = \{e_4\}$. Tenemos la pareja $\{e_{11}, e_4\}$ que genera $e^*\left(6, \{e_{11}, e_4\}\right): \langle A_1, \varnothing, A_3, \{1\}, \{0,1\}, A_6\rangle = F$, el cual no es válido, por lo que desechamos todas sus ramas.

Como no resta ninguna rama en el nodo (13) pasamos al siguiente elemento en C_1.

5.2 Consideramos el nodo (14). Entonces $\underline{e}_g = \varnothing$, pues ningún *edit* en $\underline{e}_{EM}$ entra en el campo 4, pero no en el 1. Por lo tanto desechamos todas sus ramas.

5.3 Consideramos el nodo (16). Entonces $\underline{e}_g = \{e_4\}$.

6.2 Tenemos la pareja $\{e_6, e_4\}$, la cual genera el *edit* implícito $e^*(6,\{e_6,e_4\}):\langle A_1,\{2\},\{1\},\{1\},A_5,A_6\rangle = F$, el cual es un *edit* esencialmente nuevo no redundante y corresponde en la Tabla 4.1 a e_{16}.

7.1 De los nodos (162), (163), (164) y (165) nos quedamos con (162), (163) y (164), pues los campos 2, 3 y 4 entran en el *edit* e_{16} generado en el paso anterior. Entonces $C_{16} = \{2, 3, 4\}$ y $m_{16} = 3$.

Consideramos el nodo (162). Entonces $\underline{e}_g = \{e_1\}$. Tenemos la pareja $\{e_{16}, e_1\}$ que genera $e^*(2,\{e_{16},e_1\}):\langle A_1,A_2,\varnothing,\{1\},\{0,1\},A_6\rangle = F$, el cual no es válido, por lo que desechamos todas sus ramas.

Consideramos el nodo (163). Entonces $\underline{e}_g = \{e_1\}$. Tenemos la pareja $\{e_{16}, e_1\}$ que genera $e^*(3,\{e_{16},e_1\}):\langle A_1,\varnothing,A_3,\{1\},\{0,1\},A_6\rangle = F$, el cual no es válido, por lo que desechamos todas sus ramas.

Consideramos el nodo (164). Entonces $\underline{e}_g = \varnothing$, por lo que desechamos todas sus ramas.

Como no resta ninguna rama en el nodo (16) y como no resta ningún elemento en C_1 vamos al paso 8.

8. Agregamos e_6, e_{11} y e_{16} a $\underline{e}_E$ por lo que $\underline{e}_E = \{e_1, e_2, e_3, e_4, e_5, e_6, e_{11}, e_{16}\}$.

La Tabla 6.1 resume la aplicación del algoritmo a los demás árboles del BCC2. Finalmente, la Tabla 6.2 nos muestra los *edits* implícitos esencialmente nuevos no redundantes que se obtuvieron en el ejemplo, así como sus *edits* contribuyentes y el nodo en el que se obtuvieron.

Existen dos diferencias principales entre el Algoritmo WW y el GKL1:

- Con el Algoritmo WW, cada árbol del BCC2 requiere los nodos de la forma (ijk...), $i, j, k \in \{1,...,n\}$.
- Con los pasos 4, 5 y 6 del Algoritmo WW restringimos el número de *edits* que pasan al siguiente ciclo de generación de *edits*; con el Algoritmo GKL1 no hay restricción y el número de *edits* puede crecer exponencialmente.

Aún así los cálculos en cada nodo son todavía exponenciales en el número de *edits* como ocurre en el Algoritmo GKL1.

Nodo	e_I	C_i y m_i	e_g	Pareja	*Edits* implícitos	EENNR[1]
(2)	$\{e_1, e_3, e_4\}$			$\{e_1, e_3\}$	$e^*(2,\{e_1,e_3\}):\langle\{0\},A_2,\{0\},\{1,2,3\},\{0,1\},A_6\rangle = F$	e_7
				$\{e_1, e_4\}$	$e^*(2,\{e_1,e_4\}):\langle A_1,A_2,\{0\},A_4,\{0,1\},\{0,1\}\rangle = F$	e_8
				$\{e_3, e_4\}$	$e^*(2,\{e_3,e_4\}):\langle\{0\},A_2,A_3,\{1,2,3\},A_5,\{0,1\}\rangle = F$	e_9
(21)		$C_2 = \{1, 3, 4, 5, 6\}$ $m_2 = 5$	$\{e_2, e_5\}$	$\{e_7, e_2\}$	$e^*(1,\{e_7,e_2\}):\langle A_1,A_2,\varnothing,\{1\},\{0,1\},\{2,3\}\rangle = F$	
				$\{e_7, e_5\}$	$e^*(1,\{e_7,e_5\}):\langle A_1,A_2,\{0\},\varnothing,\{1\},A_6\rangle = F$	
				$\{e_8, e_2\}$	$e^*(1,\{e_8,e_2\}):\langle A_1,A_2,\varnothing,\{0,1\},\{0,1\},\varnothing\rangle = F$	
				$\{e_8, e_5\}$	$e^*(1,\{e_8,e_5\}):\langle A_1,A_2,\{0\},\{0\},\{1\},\{0,1\}\rangle = F$	Dominado por e_8
				$\{e_9, e_2\}$	$e^*(1,\{e_9,e_2\}):\langle A_1,A_2,\{1\},\{1\},A_5,\varnothing\rangle = F$	
				$\{e_9, e_5\}$	$e^*(1,\{e_9,e_5\}):\langle A_1,A_2,A_3,\varnothing,\{1,2\},\{0,1\}\rangle = F$	
(23)			$\{e_2\}$	$\{e_7, e_2\}$	$e^*(3,\{e_7,e_2\}):\langle\varnothing,A_2,A_3,\{1\},\{0,1\},\{2,3\}\rangle = F$	
				$\{e_8, e_2\}$	$e^*(3,\{e_8,e_2\}):\langle\{1\},A_2,A_3,\{0,1\},\{0,1\},\varnothing\rangle = F$	
				$\{e_9, e_2\}$	$e^*(3,\{e_9,e_2\}):\langle\varnothing,A_2,A_3,\{1\},A_5,\varnothing\rangle = F$	
(24)		$C_2 = \{1, 3, 4, 5, 6\}$ $m_2 = 5$	$\{e_2, e_5\}$	$\{e_7, e_2\}$	$e^*(4,\{e_7,e_2\}):\langle\varnothing,A_2,\varnothing,A_4,\{0,1\},\{2,3\}\rangle = F$	
				$\{e_7, e_5\}$	$e^*(4,\{e_7,e_5\}):\langle\varnothing,A_2,\{0\},A_4,\{1\},A_6\rangle = F$	
				$\{e_8, e_2\}$	$e^*(4,\{e_8,e_2\}):\langle\{1\},A_2,\varnothing,A_4,\{0,1\},\varnothing\rangle = F$	
				$\{e_8, e_5\}$	$e^*(4,\{e_8,e_5\}):\langle\{1\},A_2,\{0\},A_4,\{1\},\{0,1\}\rangle = F$	Dominado por e_8
				$\{e_9, e_2\}$	$e^*(4,\{e_9,e_2\}):\langle\varnothing,A_2,\{1\},A_4,A_5,\varnothing\rangle = F$	
				$\{e_9, e_5\}$	$e^*(4,\{e_9,e_5\}):\langle\varnothing,A_2,A_3,A_4,\{1,2\},\{0,1\}\rangle = F$	
(25)			$\{e_5\}$	$\{e_7, e_5\}$	$e^*(5,\{e_7,e_5\}):\langle\varnothing,A_2,\{0\},\varnothing,A_5,A_6\rangle = F$	
				$\{e_8, e_5\}$	$e^*(5,\{e_8,e_5\}):\langle A_1,A_2,\{0\},A_4,\{1\},\{0,1\}\rangle = F$	Dominado por e_8
				$\{e_9, e_5\}$	$e^*(5,\{e_9,e_5\}):\langle\varnothing,A_2,A_3,\varnothing,A_5,\{0,1\}\rangle = F$	
(26)			$\{e_2\}$	$\{e_7, e_2\}$	$e^*(6,\{e_7,e_2\}):\langle\varnothing,A_2,\varnothing,\{1\},\{0,1\},A_6\rangle = F$	
				$\{e_8, e_2\}$	$e^*(6,\{e_8,e_2\}):\langle\{1\},A_2,\varnothing,\{0,1\},\{0,1\},A_6\rangle = F$	
				$\{e_9, e_2\}$	$e^*(6,\{e_9,e_2\}):\langle\varnothing,A_2,\{1\},\{1\},A_5,A_6\rangle = F$	

Tabla 6.1. *Resultados de la aplicación del Algoritmo WW a los árboles correspondientes a los campos 2, 3, 4, 5 y 6.* [1] edit *esencialmente nuevo no redundante.*

Nodo	$\underline{e_l}$	C_i y m_i	$\underline{e_g}$	Pareja	*Edits* implícitos	EENNR
(3)	$\{e_1,e_2\}$			$\{e_1,e_2\}$	$e^*(3,\{e_1,e_2\}):\langle\{1\},\{0,1\},A_3,\{0,1\},\{0,1\},\{2,3\}\rangle = F$	e_{10}
(31)		$C_3 = \{1, 2, 4, 5, 6\}$ $m_3 = 5$	$\{e_3, e_5\}$	$\{e_{10},e_3\}$	$e^*(1,\{e_{10},e_3\}):\langle A_1,\{1\},A_3,\{1\},\{0,1\},\{2,3\}\rangle = F$	Dominado por e_{11}
				$\{e_{10},e_5\}$	$e^*(1,\{e_{10},e_5\}):\langle\{1\},\{0,1\},A_3,\{0\},\{0,1\},\{2,3\}\rangle = F$	
(32)			$\{e_3,e_4\}$	$\{e_{10},e_3\}$	$e^*(2,\{e_{10},e_3\}):\langle\varnothing,A_2,A_3,\{1\},\{0,1\},\{2,3\}\rangle = F$	
				$\{e_{10},e_4\}$	$e^*(2,\{e_{10},e_4\}):\langle A_1,\{0\},A_3,\{0,1\},\{0,1\},\varnothing\rangle = F$	
(34)			$\{e_3,e_5\}$	$\{e_{10},e_3\}$	$e^*(4,\{e_{10},e_3\}):\langle\varnothing,\{1\},A_3,A_4,\{0,1\},\{2,3\}\rangle = F$	
				$\{e_{10},e_5\}$	$e^*(4,\{e_{10},e_5\}):\langle\{0\},\{0,1\},A_3,\{0,1\},\{1\},\{2,3\}\rangle = F$	
(35)			$\{e_5\}$	$\{e_{10},e_5\}$	$e^*(5,\{e_{10},e_5\}):\langle\{1\},\{0,1\},A_3,\{0\},A_5,\{2,3\}\rangle = F$	e_{14}
(351)		$C_{35} = \{1, 2, 4, 6\}$ $m_{35} = 4$	$\{e_3\}$	$\{e_{14},e_3\}$	$e^*(1,\{e_{14},e_3\}):\langle A_1,\{1\},A_3,\varnothing,A_5,\{2,3\}\rangle = F$	
(352)			$\{e_3,e_4\}$	$\{e_{14},e_3\}$	$e^*(2,\{e_{14},e_3\}):\langle\varnothing,A_2,A_3,\varnothing,A_5,\{2,3\}\rangle = F$	
				$\{e_{14},e_4\}$	$e^*(2,\{e_{14},e_4\}):\langle\{1\},A_2,A_3,\{0\},A_5,\varnothing\rangle = F$	
(354)			$\{e_3\}$	$\{e_{14},e_3\}$	$e^*(4,\{e_{14},e_3\}):\langle\varnothing,\{1\},A_3,A_4,A_5,\{2,3\}\rangle = F$	
(356)			$\{e_4\}$	$\{e_{14},e_4\}$	$e^*(6,\{e_{14},e_4\}):\langle\{1\},\{0\},A_3,\{0\},A_5,A_6\rangle = F$	e_{18}
(3561)		$C_{356} = \{1, 2, 4\}$ $m_{356} = 3$	$\{e_3\}$	$\{e_{18},e_3\}$	$e^*(1,\{e_{18},e_3\}):\langle A_1,\varnothing,A_3,\varnothing,A_5,A_6\rangle = F$	
(3562)			$\{e_3\}$	$\{e_{18},e_3\}$	$e^*(2,\{e_{18},e_3\}):\langle\varnothing,A_2,A_3,\varnothing,A_5,A_6\rangle = F$	
(3564)			$\{e_3\}$	$\{e_{18},e_3\}$	$e^*(4,\{e_{18},e_3\}):\langle\varnothing,A_2,A_3,\varnothing,A_5,A_6\rangle = F$	

Tabla 6.1. *Resultados de la aplicación del Algoritmo WW a los árboles correspondientes a los campos 2, 3, 4, 5 y 6. (Continuación)*

Nodo	e_f	C_i y m_i	e_g	Pareja	*Edits* implícitos	EENNR
(36)			$\{e_4\}$	$\{e_{10}, e_4\}$	$e^*(6, \{e_{10}, e_4\}) : \langle \{1\}, \{0\}, A_3, \{0,1\}, \{0,1\}, A_6 \rangle = F$	e_{17}
(361)		$C_{36} = \{1, 2, 4, 5\}$ $m_{36} = 4$	$\{e_3, e_5\}$	$\{e_{17}, e_3\}$	$e^*(1, \{e_{17}, e_3\}) : \langle A_1, \varnothing, A_3, \{1\}, \{0,1\}, A_6 \rangle = F$	
				$\{e_{17}, e_5\}$	$e^*(1, \{e_{17}, e_5\}) : \langle \{1\}, \{0\}, A_3, \{0\}, \{1\}, A_6 \rangle = F$	Dominado por e_{17}
(362)			$\{e_3\}$	$\{e_{17}, e_3\}$	$e^*(2, \{e_{17}, e_3\}) : \langle \varnothing, A_2, A_3, \{1\}, \{0,1\}, A_6 \rangle = F$	
(364)			$\{e_3, e_5\}$	$\{e_{17}, e_3\}$	$e^*(4, \{e_{17}, e_3\}) : \langle \varnothing, \varnothing, A_3, A_4, \{0,1\}, A_6 \rangle = F$	
				$\{e_{17}, e_5\}$	$e^*(4, \{e_{17}, e_5\}) : \langle \{1\}, \{0\}, A_3, \{0,1\}, \{1\}, A_6 \rangle = F$	Dominado por e_{17}
(365)			$\{e_5\}$	$\{e_{17}, e_5\}$	$e^*(5, \{e_{17}, e_5\}) : \langle \{1\}, \{0\}, A_3, \{0\}, A_5, A_6 \rangle = F$	Dominado por e_{18}
(4)	$\{e_2, e_3, e_5\}$			$\{e_2, e_3\}$	$e^*(4, \{e_2, e_3\}) : \langle \varnothing, \{1,2\}, \{1\}, A_4, A_5, \{2,3\} \rangle = F$	
				$\{e_2, e_5\}$	$e^*(4, \{e_2, e_5\}) : \langle \{1\}, A_2, \{1\}, \{0,1\}, \{1,2\}, \{2,3\} \rangle = F$	Dominado por e_2
				$\{e_3, e_5\}$	$e^*(4, \{e_3, e_5\}) : \langle \varnothing, \{1,2\}, A_3, A_4, \{1,2\}, A_6 \rangle = F$	
(5)	$\{e_2, e_5\}$			$\{e_1, e_5\}$	$e^*(5, \{e_1, e_5\}) : \langle \{1\}, \{0,1\}, \{0\}, \{0\}, A_5, A_6 \rangle = F$	e_{12}
(51)		$C_5 = \{1, 2, 3, 4\}$ $m_5 = 4$	$\{e_2, e_3\}$	$\{e_{12}, e_2\}$	$e^*(1, \{e_{12}, e_2\}) : \langle \{1\}, \{0,1\}, \varnothing, \{0\}, A_5, A_6 \rangle = F$	
				$\{e_{12}, e_3\}$	$e^*(1, \{e_{12}, e_3\}) : \langle A_1, \{1\}, \{0\}, \varnothing, A_5, A_6 \rangle = F$	
(52)			$\{e_3, e_4\}$	$\{e_{12}, e_3\}$	$e^*(2, \{e_{12}, e_3\}) : \langle \varnothing, A_2, \{0\}, \varnothing, A_5, A_6 \rangle = F$	
				$\{e_{12}, e_4\}$	$e^*(2, \{e_{12}, e_4\}) : \langle \{1\}, A_2, \{0\}, \{0\}, A_5, \{0,1\} \rangle = F$	e_{13}
(521)			$\{e_2\}$	$\{e_{13}, e_2\}$	$e^*(1, \{e_{13}, e_2\}) : \langle \{1\}, A_2, \varnothing, \{0\}, A_5, \varnothing \rangle = F$	

Tabla 6.1. *Resultados de la aplicación del Algoritmo WW a los árboles correspondientes a los campos 2, 3, 4, 5 y 6. (Continuación)*

Nodo	e_I	C_i y m_i	e_g	Pareja	*Edits* implícitos	EENNR
(523)		$C_{52} = \{1, 3, 4, 6\}$ $m_{52} = 4$	$\{e_2\}$	$\{e_{13}, e_2\}$	$e^*(3, \{e_{13}, e_2\}) : \langle \{1\}, A_2, A_3, \{0\}, A_5, \varnothing \rangle = F$	
(524)			$\{e_2\}$	$\{e_{13}, e_2\}$	$e^*(4, \{e_{13}, e_2\}) : \langle \{1\}, A_2, \varnothing, \{0,1\}, A_5, \varnothing \rangle = F$	
(526)			$\{e_2\}$	$\{e_{13}, e_2\}$	$e^*(6, \{e_{13}, e_2\}) : \langle \{1\}, A_2, \varnothing, \{0\}, A_5, A_6 \rangle = F$	
(53)		$C_5 = \{1, 2, 3, 4\}$ $m_5 = 4$	$\{e_2\}$	$\{e_{12}, e_2\}$	$e^*(3, \{e_{12}, e_2\}) : \langle \{1\}, \{0,1\}, A_3, \{0\}, A_5, \{2,3\} \rangle = F$	Dominado por e_{14}
(54)			$\{e_2, e_3\}$	$\{e_{12}, e_2\}$	$e^*(4, \{e_{12}, e_2\}) : \langle \{1\}, \{0,1\}, \varnothing, \{0,1\}, A_5, \{2,3\} \rangle = F$	
				$\{e_{12}, e_3\}$	$e^*(4, \{e_{12}, e_3\}) : \langle \varnothing, \{1\}, \{0\}, A_4, A_5, A_6 \rangle = F$	
(6)	$\{e_2, e_4\}$			$\{e_2, e_4\}$	$e^*(6, \{e_2, e_4\}) : \langle \{1\}, \{0,2\}, \{1\}, \{0,1\}, A_5, A_6 \rangle = F$	e_{15}
(61)		$C_6 = \{1, 2, 3, 4\}$ $m_6 = 4$	$\{e_3, e_5\}$	$\{e_{15}, e_3\}$	$e^*(1, \{e_{15}, e_3\}) : \langle A_1, \{2\}, \{1\}, \{1\}, A_5, A_6 \rangle = F$	Dominado por e_{16}
				$\{e_{15}, e_5\}$	$e^*(1, \{e_{15}, e_5\}) : \langle \{1\}, \{0,2\}, \{1\}, \{0\}, \{1,2\}, A_6 \rangle = F$	
(62)			$\{e_2, e_3\}$	$\{e_{15}, e_1\}$	$e^*(2, \{e_{15}, e_1\}) : \langle \{1\}, A_2, \varnothing, \{0,1\}, \{0,1\}, A_6 \rangle = F$	
				$\{e_{15}, e_3\}$	$e^*(2, \{e_{15}, e_3\}) : \langle \varnothing, A_2, \{1\}, \{1\}, A_5, A_6 \rangle = F$	
(63)			$\{e_1\}$	$\{e_{15}, e_1\}$	$e^*(3, \{e_{15}, e_1\}) : \langle \{1\}, \{0\}, A_3, \{0,1\}, \{0,1\}, A_6 \rangle = F$	Dominado por e_{17}.
(64)			$\{e_3, e_5\}$	$\{e_{15}, e_3\}$	$e^*(4, \{e_{15}, e_3\}) : \langle \varnothing, \{2\}, \{1\}, A_4, A_5, A_6 \rangle = F$	
				$\{e_{15}, e_5\}$	$e^*(4, \{e_{15}, e_5\}) : \langle \{1\}, \{0,2\}, \{1\}, \{0,1\}, \{1,2\}, A_6 \rangle = F$	

Tabla 6.1. *Resultados de la aplicación del Algoritmo WW a los árboles correspondientes a los campos 2, 3, 4, 5 y 6. (Continuación)*

Nodo	*Edits* Esencialmente Nuevos No Redundantes	*Edits* Generadores
(1)	$e_6 : \langle A_1, \{1,2\}, \{1\}, \{1\}, A_5, \{2,3\} \rangle = F$	e_2, e_3
(13)	$e_{11} : \langle A_1, \{1\}, A_3, \{1\}, \{0,1\}, \{2,3\} \rangle = F$	e_1, e_6
(16)	$e_{16} : \langle A_1, \{2\}, \{1\}, \{1\}, A_5, A_6 \rangle = F$	e_4, e_6
(2)	$e_7 : \langle \{0\}, A_2, \{0\}, \{1,2,3\}, \{0,1\}, A_6 \rangle = F$	e_1, e_3
	$e_8 : \langle A_1, A_2, \{0\}, A_4, \{0,1\}, \{0,1\} \rangle = F$	e_1, e_4
	$e_9 : \langle \{0\}, A_2, A_3, \{1,2,3\}, A_5, \{0,1\} \rangle = F$	e_3, e_4
(3)	$e_{10} : \langle \{1\}, \{0,1\}, A_3, \{0,1\}, \{0,1\}, \{2,3\} \rangle = F$	e_1, e_2
(35)	$e_{14} : \langle \{1\}, \{0,1\}, A_3, \{0\}, A_5, \{2,3\} \rangle = F$	e_5, e_{10}
(356)	$e_{18} : \langle \{1\}, \{0\}, A_3, \{0\}, A_5, A_6 \rangle = F$	e_4, e_{14}
(36)	$e_{17} : \langle \{1\}, \{0\}, A_3, \{0,1\}, \{0,1\}, A_6 \rangle = F$	e_4, e_{10}
(5)	$e_{12} : \langle \{1\}, \{0,1\}, \{0\}, \{0\}, A_5, A_6 \rangle = F$	e_1, e_5
(52)	$e_{13} : \langle \{1\}, A_2, \{0\}, \{0\}, A_5, \{0,1\} \rangle = F$	e_{12}, e_4
(6)	$e_{15} : \langle \{1\}, \{0,2\}, \{1\}, \{0,1\}, A_5, A_6 \rangle = F$	e_2, e_4

Tabla 6.2. Edits *implícitos esencialmente nuevos no redundantes del Ejemplo 6.2.*

CAPÍTULO VII
ALGORITMO BCC

Tanto el Algoritmo GKL1 como el WW no apelan a la rutina del PSC para generar *edits* implícitos. Así, un algoritmo que incorporara dicho problema es altamente deseable para reducir los tiempos de computadora en la generación de *edits*. En este capítulo se describe un nuevo algoritmo que implementa el *set covering*.

Este nuevo algoritmo reduce significativamente los cálculos para la generación de *edits* de diversas formas:

- No produce cubiertas redundantes y así evita generar *edits* redundantes en el proceso de generación de *edits*.
- Minimiza el tamaño de la matriz requerida para producir cubiertas principales.
- Utiliza operaciones binarias para cálculos más eficientes.

7.1 ALGUNOS RESULTADOS TEÓRICOS Y OBSERVACIONES SOBRE EL ALGORITMO BCC.

El Algoritmo BCC encuentra todas las cubiertas principales para el PSC dado por (4.4) – (4.6), conocido también como un PSC con matriz $G_i = \left(g^i_{jr}\right)_{n_i \times s_i}$, con j-ésima fila $g^i_{j\bullet} = [g^i_{j1}, g^i_{j2}, ..., g^i_{js_i}]$ y r-ésima columna $g^i_{\bullet r} = [g^i_{0r}, g^i_{1r}, ..., g^i_{(n_i-1)r}]^t$, donde $s_i = \#(S_i)$.

El Algoritmo BCC reduce el número de columnas en la matriz G_i en (4.5) al remover las columnas duplicadas. El siguiente es el teorema que permite realizar esto.

Teorema 7.1:

En el PSC dado por (4.4) – (4.6), si $g^i_{\bullet p} = g^i_{\bullet q}$ y $\{r_1,\ldots,r_k,p\}$ es una cubierta principal del siguiente PSC reducido:

$$\begin{aligned} &\text{Minimizar} \sum_{r\in S_i-\{q\}} x_r \\ &\text{Sujeto a} \sum_{r\in S_i-\{q\}} g^i_{jr} x_r \geq 1,\ j=0,1,\ldots,n_i-1 \\ &\qquad x_j = 0,1,\quad j=0,1,\ldots,n_i-1 \end{aligned} \tag{7.1}$$

Entonces tanto $\{r_1,\ldots,r_k,p\}$ como $\{r_1,\ldots,r_k,q\}$ son cubiertas principales del PSC dado por (4.4) – (4.6).

Demostración

Demostraremos este teorema por contradicción.

1) Supongamos que $\{r_1,\ldots,r_k,p\}$ no es cubierta principal de (4.4) lo cual implica que $\{r_1,\ldots,r_k,p\}$ es una cubierta redundante de (4.4). Entonces existe $r_j \in \{r_1,\ldots,r_k,p\}$ tal que $\{r_1,\ldots,r_k,p\}-\{r_j\}$ es una cubierta de (4.4). Como esta cubierta no contiene a q, $\{r_1,\ldots,r_k,p\}-\{r_j\}$ es una cubierta de (7.1). Pero por hipótesis $\{r_1,\ldots,r_k,p\}$ es una cubierta principal de (7.1) por lo que llegamos a una contradicción.

2) Supongamos que $\{r_1,\ldots,r_k,q\}$ no es una cubierta principal de (4.4) lo cual implica que $\{r_1,\ldots,r_k,q\}$ es una cubierta redundante de (4.4). Entonces existe $r_j \in \{r_1,\ldots,r_k,q\}$ tal que $\{r_1,\ldots,r_k,q\}-\{r_j\}$ es una cubierta de (7.1). Tenemos así dos casos:

   a) Si $r_j = q$ entonces $\{r_1,\ldots,r_k\}$ es una cubierta de (4.4). Como esta cubierta no contiene a q, $\{r_1,\ldots,r_k\}$ es una cubierta de (7.1). Pero por hipótesis $\{r_1,\ldots,r_k,p\}$ es una cubierta principal de (7.1) por lo que llegamos a una contradicción.

b) Si $r_j \in \{r_1,...,r_k\}$ entonces $\{r_1,...,r_{j-1},r_{j+1},...,r_k,q\}$ es cubierta de (4.4). Por hipótesis $g_{\bullet p} = g_{\bullet q}$, lo que implica que $\{r_1,...,r_{j-1},r_{j+1},...,r_k,p\}$ es cubierta de (7.1). Como esta cubierta no contiene a q, $\{r_1,...,r_{j-1},r_{j+1},...,r_k,p\}$ es cubierta de (4.4). Pero por hipótesis $\{r_1,...,r_k,p\}$ es una cubierta principal de (7.1) por lo que llegamos a una contradicción. ■

Aunque el número máximo de columnas diferentes es $2^{n_i}-2$, el tamaño real es mucho más pequeño debido al número limitado de *edits* en $\underline{e}_i$ (los *edits* que entran en el campo *i*). Cuando $g^i_{\bullet p} = g^i_{\bullet q}$, significa que los *edits* e_p y e_q pertenecientes a $\underline{e}_i$ tienen los mismos valores para el campo *i* entrando en ellos.

Ejemplo 7.1:

Este ejemplo está basado en el Ejemplo 4.1.

Supongamos que tenemos 6 campos con:

$A_1 = \{0,1\}$ $n_1 = 2$ $A_4 = \{0,1,2,3\}$ $n_4 = 4$
$A_2 = \{0,1,2\}$ $n_2 = 3$ $A_5 = \{0,1,2\}$ $n_5 = 3$
$A_3 = \{0,1\}$ $n_3 = 2$ $A_6 = \{0,1,2,3\}$ $n_6 = 4$

Y los siguientes *edits* explícitos:

$e_1 : \langle A_1, \{0,1\}, \{0\}, A_4, \{0,1\}, A_6 \rangle = F,$
$e_2 : \langle \{1\}, A_2, \{1\}, \{0,1\}, A_5, \{2,3\} \rangle = F,$
$e_3 : \langle \{0\}, \{1,2\}, A_3, \{1,2,3\}, A_5, A_6 \rangle = F,$
$e_4 : \langle A_1, \{0,2\}, A_3, A_4, A_5, \{0,1\} \rangle = F,$
$e_5 : \langle \{1\}, A_2, A_3, \{0\}, \{1,2\}, A_6 \rangle = F.$

Entonces e_2 y e_5 son dos *edits* diferentes en los que el campo 1 entra, pero $g^1_{\bullet 2} = g^1_{\bullet 5} = [0,\ 1]$. En este ejemplo, el PSC dado por (4.4) – (4.6) tiene a $s_i = 3$ pero sólo dos columnas distintas.

Lema 7.1:

Si $g^i_{u\bullet} \geq g^i_{v\bullet}$ (es decir, $g^i_{ur} \geq g^i_{vr}$ $\forall r \in S_i$) para alguna u y v, entonces cada cubierta de la fila v cubre a la fila u.

Demostración

Tenemos que, como $g^i_{u\bullet} \geq g^i_{v\bullet}$, por definición, $g^i_{ur} \geq g^i_{vr}\ \forall r \in S_i$. Ahora, si x es una cubierta de la fila v, entonces $\sum_{r \in S_i} g^i_{vr} x_r \geq 1$. De lo anterior tenemos que $\sum_{r \in S_i} g^i_{ur} x_r \geq \sum_{r \in S_i} g^i_{vr} x_r \geq 1$. Por lo tanto, x también será cubierta de la fila u. ■

El objetivo del Teorema 7.1 y el Lema 7.1 es reducir el tamaño de la matriz G_i y así minimizar el tamaño del BCC1 (que se generará antes de que una rutina de set-covering se aplique). Una nueva rutina de set-covering es descrita abajo para minimizar el número de nodos a ser visitados en el BCC1. La nueva rutina no se interesa en ninguna cubierta mas que en las principales.

En un BCC1, si se descubre que un nodo corresponde a una cubierta, sus ramas no serán cubiertas principales, por lo que no serán visitadas. Así, desearíamos encontrar un nodo cubierta tan cerca como fuera posible a cualquier nodo raíz. Esto minimizará el número de nodos visitados y los cálculos para encontrar una cubierta principal.

Ejemplo 7.2:

Supongamos que en la Figura 7.1 el nodo (12) es una cubierta, entonces los nodos (123), (1234) y (124) no se revisarán porque no serán cubiertas principales.

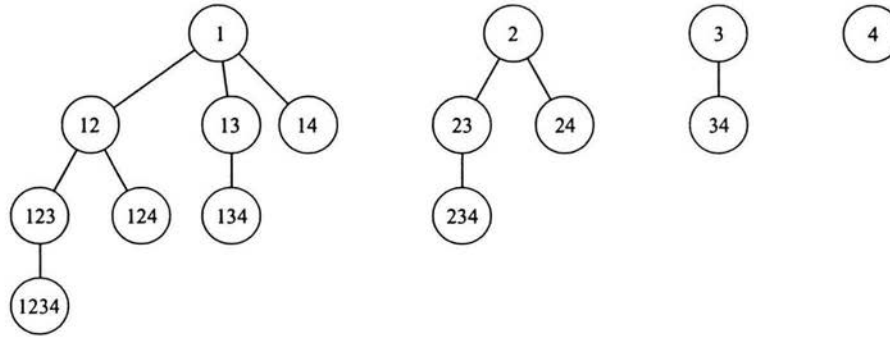


Figura 7.1. *BCC1 para el caso de cuatro campos.*

En cada nodo cubierta, las filas redundantes serán eliminadas para formar una cubierta principal. Si el número de filas redundantes es 0 ó 1, ningún cálculo adicional será necesario para encontrar una cubierta principal. Si es mayor a 1, nuevas cubiertas serán obtenidas removiendo una fila redundante cada vez. Este mismo procedimiento de remover filas redundantes es aplicado para cada nueva cubierta.

En cada nodo cubierta, si una subcubierta es una cubierta y el nodo representando la subcubierta no ha sido aún visitado, entonces el nodo y sus ramas son marcadas como visitados.

Ejemplo 7.3:

Si el nodo actual es el (123) en la Figura 7.1 y se descubre que es cubierta y (23) es una subcubierta, entonces los nodos (23) y (234) son marcados como visitados.

También, el procedimiento descrito encuentra todas las cubiertas principales.

Ejemplo 7.4:

Retomando el ejemplo anterior, el nodo (1234) no será visitado ya que el nodo (123) es una cubierta. Si el nodo (34) es una subcubierta del nodo (1234), ése será visitado a menos de que haya sido marcado como visitado antes de llegar a él.

Como se describe arriba, queremos encontrar un nodo cubierta en el BCC1 tan cerca como sea posible a cualquier nodo raíz. Esto se puede lograr realizando un paso adicional antes de que se construya el BCC1. Las columnas, $g^{i}_{\bullet r}$, son ordenadas según el número de 1's en orden descendente. El siguiente ejemplo provee información acerca de los cálculos ahorrados al ordenar las columnas.

Ejemplo 7.5:

Supongamos que se tiene una matriz

$$G_1 = \begin{pmatrix} 0 & 1 & 1 & 0 & 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 & 1 & 0 & 1 & 1 \\ 1 & 0 & 1 & 1 & 1 & 0 & 0 & 1 \\ 0 & 1 & 1 & 0 & 0 & 1 & 0 & 1 \end{pmatrix}$$

y dos matrices ordenadas con respecto al numero de 1's en las $g^{1}_{\bullet r}$'s en orden ascendente y descendente, respectivamente,

$$G_1^a = \begin{pmatrix} 0 & 0 & 0 & 0 & 1 & 1 & 1 & 0 \\ 0 & 0 & 1 & 1 & 1 & 1 & 0 & 1 \\ 1 & 0 & 1 & 1 & 0 & 0 & 1 & 1 \\ 0 & 1 & 0 & 0 & 0 & 1 & 1 & 1 \end{pmatrix} \text{ y } G_1^d = \begin{pmatrix} 1 & 1 & 0 & 0 & 0 & 1 & 0 & 0 \\ 1 & 0 & 1 & 1 & 1 & 1 & 0 & 0 \\ 0 & 1 & 1 & 1 & 1 & 0 & 1 & 0 \\ 1 & 1 & 1 & 0 & 0 & 0 & 0 & 1 \end{pmatrix}.$$

El número de nodos para el BCC1 para las tres matrices es 2^8 - 1 = 255. La siguiente tabla muestra el número de nodos visitados y de éstos los que tenían menos de dos columnas redundantes para las tres matrices:

Matriz	G_1^a	G_1	G_1^d
Número de nodos en el BCC1	255	255	255
Número de nodos visitados	70	36	21
Número de nodos visitados con menos de dos columnas redundantes	33	24	16
Porcentaje	47.1%	66.7%	76.2%

La tabla también muestra el porcentaje de nodos cubierta visitados con menos de dos columnas redundantes. Esto indica que el tiempo de computadora se reduce significativamente con G_1^d.

7.2 ALGORITMO BCC.

El algoritmo está diseñado para reducir el número de columnas necesarias para encontrar las cubiertas principales del PSC dado por (4.4) – (4.6). La reducción del número de columnas de G_i reducirá el tamaño del BCC1 correspondiente.

La siguiente es la descripción del algoritmo paso a paso tomando como campo generador el campo *i*:

*Algoritmo BCC**:

1. Remueva las columnas duplicadas de la matriz G_i en el PSC dado por (4.4) – (4.6). Sea $H_i = (h^i_{jr})_{n_i \times t_i}$ la matriz reducida, donde $t_i \le s_i$, la r-ésima columna es $h^i_{\bullet r} = [h^i_{0r}, h^i_{1r}, ..., h^i_{(n_i-1)r}]^t$ y la j-ésima fila es $h^i_{j\bullet} = [h^i_{j1}, h^i_{j2}, ..., h^i_{jt_i}]$.

2. Para cada pareja (u, v), $0 \le u < v \le n_i - 1$, verifique si $h^i_{u\bullet} \ge h^i_{v\bullet}$. Si se da la desigualdad, remueva la fila $h^i_{u\bullet}$ de H_i. Sea $B_i = (b^i_{jr})_{m_i \times t_i}$ la matriz reducida resultante, donde $m_i \le n_i$, la r-ésima columna es $b^i_{\bullet r} = [b^i_{0r}, b^i_{1r}, ..., b^i_{(m_i-1)r}]^t$ y la j-ésima fila es $b^i_{j\bullet} = [b^i_{j1}, b^i_{j2}, ..., b^i_{jt_i}]$.

3. Busque, de entre las $b^i_{j\bullet}$, $0 \le j \le m_i - 1$, aquellas que son unitarias. Sean estas filas unitarias $I^{j_1}_{r_1}, I^{j_2}_{r_2}, ..., I^{j_k}_{r_k}$, donde $I^{j_1}_{r_1} = b^i_{j_1\bullet}$ es una fila unitaria con el r_1-ésimo elemento 1 y los demás 0. En caso de que no existan filas unitarias vaya al paso 6.

4. Suponiendo que $r_1, r_2, ..., r_k$ son k números distintos, elimine de B_i las k columnas y las l filas con $g^i_{jr} = 1$, $r \in \{r_1, r_2, ..., r_k\}$ y $j \in \{j_1, j_2, ..., j_l\}$, para formar una matriz

* Con base en el algoritmo dado en Bor-Chung Chen (1998).

reducida, denotada por B_{li}, donde $j_1, j_2, ..., j_l$ son l números diferentes. Entonces B_{li} con dimensión $(m_i - l)\times(t_i - k)$ es la matriz asociada a un nuevo PSC reducido, donde $l \geq k$.

5. Si $l = m_i$, el conjunto $\{r_1, r_2, ..., r_k\}$ de B_i es la única cubierta principal. En este caso vaya al paso 12. En otro caso, este conjunto es una cubierta la cual debe verificar si es principal.

6. Reordene las columnas de B_{li} en orden decreciente basándose en el número de 1's. Sea $D_i = (d^i_{jr})_{(m_i-l)\times(t_i-k)}$ la nueva matriz con r-ésima columna $d^i_{\bullet r} = \left[d^i_{0r}, d^i_{1r}, ..., d^i_{(m_i-l-1)r}\right]$ y j-ésima fila $d^i_{j\bullet} = \left[d^i_{j1}, d^i_{j2}, ..., d^i_{j(t_i-k)}\right]^t$.

7. Construya el BCC1 para el PSC asociado a D_i. Cada nodo del BCC1 representa un conjunto de índices de *edits* de D_i. Por ejemplo, el nodo (234) en la Figura 7.1 representa el conjunto de índices de los *edits* e_2, e_3 y e_4 de D_1.

8. Recorra el BCC1. En cada nodo, verifique si el conjunto de índices de los *edits* determinan una cubierta para el nuevo PSC asociado a D_i. Si lo es vaya al paso 9. En caso contrario verifique el nodo siguiente.

9. Identifique los elementos redundantes de la cubierta. Si el número de elementos redundantes es menor a 2, el conjunto de elementos no redundantes es una cubierta principal. En otro caso obtenga nuevas subcubiertas eliminando un elemento cada vez y retome este paso para cada nueva subcubierta.

10. Elimine las ramas del nodo y vaya al paso 8 si quedan elementos en el recorrido; si no, vaya al paso 11.

11. Todas las cubiertas principales del PSC con matriz asociada D_i, si las hay, serán encontradas. Agregue la cubierta $\{r_1, r_2, ..., r_k\}$ obtenida en el paso 5 en caso de que haya sido principal para formar todas las cubiertas principales para el PSC con matriz reducida B_i.

12. Si alguna columna tiene duplicados, reemplace el índice del *edit* correspondiente a la columna con cada duplicado para formar una nueva cubierta principal al PSC con matriz G_i.

13. Una vez obtenidas todas las cubiertas principales, genere un *edit* implícito por cada una, considerando los *edits* correspondientes a ésta como contribuyentes y a i como el campo generador siguiendo el procedimiento dado por el Lema 2.1. En caso de que una cubierta principal contenga un solo elemento la cubierta se desecha.

En caso de que se generen nuevos *edits* implícitos, éstos se anexarán al conjunto de *edits* explícitos a considerarse en el siguiente PSC con campo fijo $i+1$.

Ejemplo 7.6:

Retomamos el Ejemplo 7.1. Consideremos el primer campo generador, $i = 1$. Entonces el algoritmo es el siguiente:

1. Tenemos que

$$G_1 = \begin{array}{c} \\ 0 \\ 1 \end{array}\begin{array}{c} \begin{array}{ccc} e_2 & e_3 & e_5 \end{array} \\ \begin{pmatrix} 0 & 1 & 0 \\ 1 & 0 & 1 \end{pmatrix} \end{array}$$

Con PSC asociado:

$$\begin{array}{lllll} \text{Minimizar} & x_2 & + x_3 & + x_5 & \\ \text{Sujeto a} & & x_3 & & \geq 1 \\ & x_2 & + & x_5 & \geq 1 \\ & & x_2, x_3, x_5 = 0, 1 & & \end{array}$$

En esta matriz observamos que la columna de e_2 es igual a la de e_5, por lo que removemos la columna que corresponde a e_5, obteniendo

$$H_1 = \begin{array}{c} \\ 0 \\ 1 \end{array}\begin{array}{c} \begin{array}{cc} e_2 & e_3 \end{array} \\ \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix} \end{array}$$

2. Para cada pareja (u, v), $0 \leq u < v \leq 1$, es decir, para la pareja (0,1), verificamos si $h^1_{0\bullet} \geq h^1_{1\bullet}$, pero, como no se da la desigualdad, no se elimina ninguna fila y por lo tanto $B_1 = H_1$.

3. Observamos que las filas unitarias son: $I^0_3 = (0 \quad 1) = b^1_{0\bullet}$, $I^1_2 = (1 \quad 0) = b^1_{1\bullet}$.

4. Al eliminar de B_1 las dos filas y las dos columnas (pues $b^1_{12} = 1$ y $b^1_{03} = 1$) obtenemos la matriz reducida $B_{11} = \varnothing$.

5. Como $l = m_1 = 2$, el conjunto $\{2, 3\}$ es la única cubierta principal para el PSC:

$$\begin{array}{lllll} \text{Minimizar} & x_2 & + x_3 & & \\ \text{Sujeto a} & x_2 & & & \geq 1 \\ & & x_3 & & \geq 1 \\ & x_2, x_3 = 0, 1 & & & \end{array}$$

Por lo que vamos al paso 12.

12. Como la columna $g^1_{\bullet 2} = g^1_{\bullet 5}$, se reemplaza el índice de la primera columna por el de la tercera, para formar una nueva cubierta principal para el PSC con matriz G_1 dada por: {3, 5}. Así las cubiertas principales para el campo generador $i = 1$ son: {2, 3}, {3, 5}.

13. Generamos los *edits* implícitos a partir de las cubiertas principales aplicando el Lema 2.1.

$$\{2,3\} \leftrightarrow e^*\left(1,\{e_2,e_3\}\right):\langle A_1,\{1,2\},\{1\},\{1\},A_5,\{2,3\}\rangle = F$$
$$\{3,5\} \leftrightarrow e^*\left(1,\{e_3,e_5\}\right):\langle A_1,\{1,2\},A_3,\varnothing,\{1,2\},A_6\rangle = F$$

De los cuales, el único esencialmente nuevo es $e^*\left(1,\{e_2,e_3\}\right)$ al no ser vacío en ninguno de sus campos y estar completo en el campo 1. Además, al no dominar ni ser dominado, es un *edit* implícito esencialmente nuevo no redundante que corresponde a e_6 en la Tabla 4.1.

Consideremos el segundo campo generador, $i = 2$. Entonces el algoritmo es el siguiente:

1. Tenemos que:

$$G_2 = \begin{array}{c} \\ 0 \\ 1 \\ 2 \end{array}\begin{array}{c} \begin{array}{cccc} e_1 & e_3 & e_4 & e_6 \end{array} \\ \begin{pmatrix} 1 & 0 & 1 & 0 \\ 1 & 1 & 0 & 1 \\ 0 & 1 & 1 & 1 \end{pmatrix} \end{array}$$

Con PSC asociado:

$$\begin{array}{lllllll} \text{Minimizar} & x_1 & + x_3 & + x_4 & + x_6 & & \\ \text{Sujeto a} & x_1 & + & x_4 & & & \geq 1 \\ & x_1 & + x_3 & + & x_6 & & \geq 1 \\ & & x_3 & + x_4 & + x_6 & & \geq 1 \\ & & x_1, x_3, x_4, x_6 = 0, 1 & & & & \end{array}$$

En esta matriz observamos que la columna de e_3 es igual a la de e_6, por lo que removemos la columna que corresponde a e_6, obteniendo:

$$H_2 = \begin{array}{c} \\ 0 \\ 1 \\ 2 \end{array}\begin{array}{c} \begin{array}{ccc} e_1 & e_3 & e_4 \end{array} \\ \begin{pmatrix} 1 & 0 & 1 \\ 1 & 1 & 0 \\ 0 & 1 & 1 \end{pmatrix} \end{array}$$

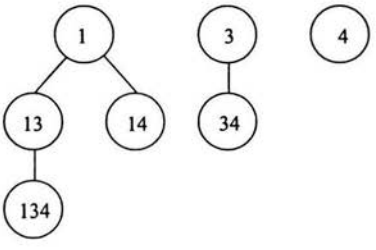
2. Para cada pareja (u, v), $0 \leq u < v \leq 2$, es decir, para las parejas (0,1), (0, 2), (1,2), se verifica si $h^2_{u\bullet} \geq h^2_{v\bullet}$; pero como no se da la desigualdad en ninguna pareja, $B_2 = H_2$.

3. Observamos que ninguna $b^2_{j\bullet}$ es unitaria ya que todas tienen más de un 1. Vamos entonces al paso 6.

6. Como la matriz B_2 ya está ordenada de forma decreciente, $D_2 = B_2$.

7. Construimos el BCC1 para el nuevo PSC asociado:

$$
\begin{array}{lllll}
\text{Minimizar} & x_1 + x_3 + x_4 & & \\
\text{Sujeto a} & x_1 + & x_4 & \geq 1 \\
& x_1 + x_3 & & \geq 1 \\
& x_3 + x_4 & & \geq 1 \\
& x_1, x_3, x_4 = 0, 1 & &
\end{array}
$$



8. Consideramos el nodo (1). Entonces la cubierta correspondiente es {1}, la cual no es una cubierta del PSC, por lo que pasamos al siguiente nodo.

   Consideramos el nodo (13). Entonces la cubierta correspondiente es {1, 3}, la cual es una cubierta del PSC.

9. Como {1, 3} no tiene elementos redundantes, tenemos que es una cubierta principal.

10. Eliminamos el nodo (134), que es rama del (13).

8'. Consideramos el nodo (14). Entonces la cubierta correspondiente es {1, 4}, la cual es una cubierta del PSC.

9'. Como {1, 4} no tiene elementos redundantes, tenemos que es una cubierta principal.

10'. Como el nodo (14) no tiene ramas vamos al paso 8''.

8''. Consideramos el nodo (3). Entonces la cubierta correspondiente es {3}, la cual no es una cubierta del PSC, por lo que pasamos al siguiente nodo.

Consideramos el nodo (34). Entonces la cubierta correspondiente es {3, 4}, la cual es una cubierta del PSC.

9''. Como {3, 4} no tiene elementos redundantes, tenemos que es una cubierta principal.

10''. Como el nodo (34) no tiene ramas vamos al paso 8'''.

8'''. Consideramos el nodo (4). Entonces la cubierta correspondiente es {4}, la cual no es una cubierta del PSC.

11. Como $D_2 = B_2$, vamos al paso 12.

12. De los pasos anteriores obtuvimos las cubiertas principales: {1, 3}, {1, 4}, {3, 4}.

Como $g_{\bullet 3} = g_{\bullet 6}$, formamos nuevas cubiertas principales al PSC con matriz G_2: {1, 6}, {4, 6}.

Por lo tanto las cubiertas principales para el campo generador i = 2 son: {1, 3}, {1, 4}, {1, 6}, {3, 4}, {4, 6}.

13. Generamos los *edits* implícitos a partir de las cubiertas principales aplicando el Lema 2.1.

$$\{1,3\} \leftrightarrow e^*\left(2,\{e_1,e_3\}\right):\langle\{0\},A_2,\{0\},\{1,2,3\},\{0,1\},A_6\rangle = F$$
$$\{1,4\} \leftrightarrow e^*\left(2,\{e_1,e_4\}\right):\langle A_1,A_2,\{0\},A_4,\{0,1\},\{0,1\}\rangle = F$$
$$\{3,4\} \leftrightarrow e^*\left(2,\{e_3,e_4\}\right):\langle\{0\},A_2,A_3,\{1,2,3\},A_5,\{0,1\}\rangle = F$$
$$\{1,6\} \leftrightarrow e^*\left(2,\{e_1,e_6\}\right):\langle A_1,A_2,\varnothing,\{1\},\{0,1\},\{2,3\}\rangle = F$$
$$\{4,6\} \leftrightarrow e^*\left(2,\{e_4,e_6\}\right):\langle A_1,A_2,\{1\},\{1\},A_5,\varnothing\rangle = F$$

De los cuales, los esencialmente nuevos no redundantes son $e^*\left(2,\{e_1,e_3\}\right)$, $e^*\left(2,\{e_1,e_4\}\right)$ y $e^*\left(2,\{e_3,e_4\}\right)$ que corresponden, respectivamente, a e_7, e_8 y e_9 en la Tabla 4.1.

Consideremos el campo generador i = 3. Entonces el algoritmo es el siguiente:

1. Tenemos que:

$$G_3 = \begin{array}{c} \\ 0 \\ 1 \end{array}\begin{array}{c} \begin{array}{ccccc} e_1 & e_2 & e_6 & e_7 & e_8 \end{array} \\ \begin{pmatrix} 1 & 0 & 0 & 1 & 1 \\ 0 & 1 & 1 & 0 & 0 \end{pmatrix} \end{array}$$

Con PSC asociado:

$$\begin{array}{lllllll} \text{Minimizar} & x_1 & + x_2 & + x_6 & + x_7 & + x_8 & \\ \text{Sujeto a} & x_1 & + & & x_7 & + x_8 & \geq 1 \\ & & x_2 & + x_6 & & & \geq 1 \\ & & x_1, x_2, x_6, x_7, x_8 = 0, 1 & & & & \end{array}$$

En esta matriz observamos que la columna de e_1 es igual a la de e_7 y e_8 y la columna e_2 es igual a e_6 por lo que removemos las columnas que corresponden a e_6, e_7 y e_8 obteniendo

$$H_3 = \begin{array}{c} \\ 0 \\ 1 \end{array}\begin{array}{c} \begin{array}{cc} e_1 & e_2 \end{array} \\ \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix} \end{array}$$

2. Para cada pareja (u, v), $0 \leq u < v \leq 1$, es decir, para la pareja (0,1) verificamos si $h^3_{0\bullet} \geq h^3_{1\bullet}$, pero, como no se da la desigualdad, no se elimina ninguna fila y por lo tanto $B_3 = H_3$.

3. Observamos que las filas unitarias son: $I^0_1 = (1 \quad 0) = b^3_{0\bullet}$, $I^1_2 = (0 \quad 1) = b^3_{1\bullet}$.

4. Al eliminar de B_3 las dos filas y las dos columnas (pues $b^3_{01} = 1$ y $b^3_{12} = 1$) obtenemos la matriz reducida $B_{13} = \varnothing$.

5. Como $l = m_3 = 2$, el conjunto $\{1, 2\}$ es la única cubierta principal para el PSC:

$$\begin{array}{lllll} \text{Minimizar} & x_1 & + x_2 & & \\ \text{Sujeto a} & x_1 & & & \geq 1 \\ & & x_2 & & \geq 1 \\ & x_1, x_2 = 0, 1 & & & \end{array}$$

Por lo que vamos al paso 12.

12. Como $g^3_{\bullet 1} = g^3_{\bullet 7} = g^3_{\bullet 8}$ y $g^3_{\bullet 2} = g^3_{\bullet 6}$, formamos nuevas cubiertas principales al PSC con matriz G_3: {1, 6}, {2, 7}, {2, 8}, {6, 7}, {6, 8}.

Por lo tanto, las cubiertas principales para el campo generador i = 3 son: {1, 2}, {1, 6}, {2, 7}, {2, 8}, {6, 7}, {6, 8}.

13. Generamos los *edits* implícitos a partir de las cubiertas principales aplicando el Lema 2.1.

$$\{1,2\} \leftrightarrow e^*\left(3,\{e_1,e_2\}\right):\langle\{1\},\{0,1\},A_3,\{0,1\},\{0,1\},\{2,3\}\rangle = F$$
$$\{2,7\} \leftrightarrow e^*\left(3,\{e_2,e_7\}\right):\langle\varnothing,A_2,A_3,\{1\},\{0,1\},\{2,3\}\rangle = F$$
$$\{2,8\} \leftrightarrow e^*\left(3,\{e_2,e_8\}\right):\langle\{1\},A_2,A_3,\{0,1\},\{0,1\},\varnothing\rangle = F$$
$$\{1,6\} \leftrightarrow e^*\left(3,\{e_1,e_6\}\right):\langle A_1,\{1\},A_3,\{1\},\{0,1\},\{2,3\}\rangle = F$$
$$\{6,7\} \leftrightarrow e^*\left(3,\{e_6,e_7\}\right):\langle\{0\},\{1,2\},A_3,\{1\},\{0,1\},\{2,3\}\rangle = F$$
$$\{6,8\} \leftrightarrow e^*\left(3,\{e_6,e_8\}\right):\langle A_1,\{1,2\},A_3,\{1\},\{0,1\},\varnothing\rangle = F$$

De los cuales, los esencialmente nuevos no redundantes son $e^*\left(3,\{e_1,e_2\}\right)$ y $e^*\left(3,\{e_1,e_6\}\right)$ que corresponden, respectivamente, a e_{10} y e_{11} en la Tabla 4.1.

Consideremos el campo generador $i = 4$. Entonces el algoritmo es el siguiente:

1. Tenemos que:

$$G_4 = \begin{array}{c} \\ 0 \\ 1 \\ 2 \\ 3 \end{array}\begin{array}{c} \begin{array}{cccccccc} e_2 & e_3 & e_5 & e_6 & e_7 & e_9 & e_{10} & e_{11} \end{array} \\ \begin{pmatrix} 1 & 0 & 1 & 0 & 0 & 0 & 1 & 0 \\ 1 & 1 & 0 & 1 & 1 & 1 & 1 & 1 \\ 0 & 1 & 0 & 0 & 1 & 1 & 0 & 0 \\ 0 & 1 & 0 & 0 & 1 & 1 & 0 & 0 \end{pmatrix} \end{array}$$

Con PSC asociado:

$$\begin{array}{lcccccccccccl} \text{Minimizar} & x_2 & + & x_3 & + & x_5 & + & x_6 & + & x_7 & + & x_9 & + & x_{10} & + x_{11} & \\ \text{Sujeto a} & x_2 & + & & & x_5 & + & & & & & & & x_{10} & & \geq 1 \\ & x_2 & + & x_3 & + & & & x_6 & + & x_7 & + & x_9 & + & x_{10} & + x_{11} & \geq 1 \\ & & & x_3 & + & & & & & x_7 & + & x_9 & & & & \geq 1 \\ & & & & & x_2, x_3, x_5, x_6, x_7, x_9, x_{10}, x_{11} = 0, 1 & & & & & & & & & & \end{array}$$

En esta matriz observamos que la columna de e_2 es igual a la de e_{10}, la de e_3 es igual a la de e_7 y e_9 y la de e_6 es igual a la de e_{11} por lo que removemos las columnas que corresponden a e_7, e_9, e_{10} y e_{11} obteniendo

$$H_4 = \begin{array}{c} \\ 0 \\ 1 \\ 2 \\ 3 \end{array}\begin{array}{c} \begin{array}{cccc} e_2 & e_3 & e_5 & e_6 \end{array} \\ \begin{pmatrix} 1 & 1 & 0 & 0 \\ 0 & 1 & 1 & 1 \\ 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \end{pmatrix} \end{array}$$

2. Para cada pareja (u, v), $0 \le u < v \le 3$, es decir, para las parejas (0,1), (0,2), (0,3), (1,2), (1,3) y (2,3), verificamos si $h^4_{u\bullet} \ge h^4_{v\bullet}$. Como $h^4_{1\bullet} \ge h^4_{2\bullet}$ y $h^4_{2\bullet} \ge h^4_{3\bullet}$, se eliminan las filas $h^4_{1\bullet}$ y $h^4_{2\bullet}$ y por lo tanto:

$$B_4 = \begin{array}{c} \\ 0 \\ 3 \end{array}\begin{array}{c} \begin{array}{cccc} e_2 & e_3 & e_5 & e_6 \end{array} \\ \begin{pmatrix} 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 \end{pmatrix} \end{array}$$

3. Observamos que la fila unitaria es: $I^3_3 = (0 \quad 1 \quad 0 \quad 0) = b^4_{3\bullet}$.

4. Al eliminar de B_4 la fila e_3 y la columna 3 obtenemos la matriz reducida

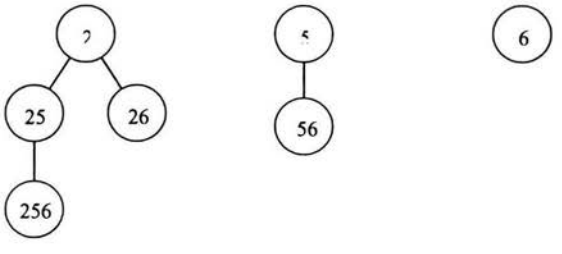
$$B_{14} = \begin{array}{c} \\ 0 \end{array}\begin{array}{c} \begin{array}{ccc} e_2 & e_5 & e_6 \end{array} \\ \begin{pmatrix} 1 & 1 & 0 \end{pmatrix} \end{array}$$

5. Como $l = 1 \ne m_4 = 2$, $\{2, 3, 5, 6\}$ es una cubierta, pero no es principal por lo que se descarta.

6. Como la matriz B_{14} ya está ordenada de manera decreciente, $D_4 = B_{14}$.

7. Construimos el BCC1 para el nuevo PSC reducido:

$$\begin{array}{lll} \text{Minimizar} & x_2 + x_5 + x_6 & \\ \text{Sujeta a} & x_2 + x_5 & \ge 1 \\ & x_2, x_5, x_6 = 0, 1 & \end{array}$$



8. Consideramos el nodo (2). Entonces la cubierta correspondiente es $\{2\}$, la cual es una cubierta del PSC.

9. Como $\{2\}$ no tiene elementos redundantes, es una cubierta principal.

10. Eliminamos los nodos (25), (26) y (256), que son ramas de (2).

8'. Consideramos el nodo (5). Entonces la cubierta correspondiente es {5}, la cual es una cubierta del PSC.

9'. Como {5} no tiene elementos redundantes, es una cubierta principal.

10'. Eliminamos el nodo (56), que es rama de (5).

8''. Consideramos el nodo (6). Entonces la cubierta correspondiente es {6}, la cual no es una cubierta del PSC.

11. Tenemos las cubiertas principales {2}, {5}.

12. Como la columna $g^4_{\bullet 2} = g^4_{\bullet 10}$, formamos una nueva cubierta principal al PSC con matriz G_4: {10}.

Por lo tanto, las cubiertas principales para el campo generador $i = 4$ son: {2}, {5}, {10}.

13. No se genera ningún *edit* implícito, pues las cubiertas están formadas por un solo elemento.

Consideremos el campo generador $i = 5$. Entonces el algoritmo es el siguiente:

1. Tenemos que:

$$G_5 = \begin{array}{c} \\ 0 \\ 1 \\ 2 \end{array} \begin{array}{c} \begin{array}{cccccc} e_1 & e_5 & e_7 & e_8 & e_{10} & e_{11} \end{array} \\ \begin{pmatrix} 1 & 0 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 & 1 \\ 0 & 1 & 0 & 0 & 0 & 0 \end{pmatrix} \end{array}$$

Con PSC asociado:

$$\begin{array}{lcccccccccl} \text{Minimizar} & x_1 & + & x_5 & + & x_7 & + & x_8 & + & x_{10} & + x_{11} & \\ \text{Sujeto a} & x_1 & + & & & x_7 & + & x_8 & + & x_{10} & + x_{11} & \geq 1 \\ & x_1 & + & x_5 & + & x_7 & + & x_8 & + & x_{10} & + x_{11} & \geq 1 \\ & & & x_5 & & & & & & & & \geq 1 \\ & & & & & x_1, x_5, x_7, x_8, x_{10}, x_{11} = 0, 1 & & & & & & \end{array}$$

En esta matriz observamos que la columna de e_1 es igual a las de e_7, e_8, e_{10}, e_{11}, por lo que removemos las columnas que corresponden a e_7, e_8, e_{10} y e_{11}, obteniendo:

$$H_5 = \begin{array}{c} \\ 0 \\ 1 \\ 2 \end{array} \begin{array}{c} \begin{array}{cc} e_1 & e_5 \end{array} \\ \begin{pmatrix} 1 & 0 \\ 1 & 1 \\ 0 & 1 \end{pmatrix} \end{array}$$

2. Para cada pareja (u, v), $0 \le u < v \le 2$, es decir, para las parejas (0,1), (0,2) y (1,2), verificamos si $h^5_{u\bullet} \ge h^5_{v\bullet}$, y como la desigualdad se da para $h^5_{1\bullet} \ge h^5_{2\bullet}$, se elimina la columna $h^5_{1\bullet}$ y por lo tanto:

$$B_5 = \begin{array}{c} \\ 0 \\ 2 \end{array} \begin{array}{c} \begin{array}{cc} e_1 & e_5 \end{array} \\ \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix} \end{array}$$

3. Observamos que las filas unitarias son: $I^0_1 = (1 \quad 0) = b^5_{0\bullet}$, $I^2_5 = (0 \quad 1) = b^5_{2\bullet}$.

4. Al eliminar de B_5 las dos filas y las dos columnas (pues $b^5_{01} = 1$ y $b^5_{25} = 1$) obtenemos la matriz reducida $B_{15} = \varnothing$.

5. Como $l = m_5 = 2$, el conjunto $\{1,5\}$ es la única cubierta principal para el PSC:

$$\begin{array}{lll} \text{Minimizar} & x_1 + x_5 & \\ \text{Sujeto a} & x_1 & \ge 1 \\ & x_5 & \ge 1 \\ & x_1, x_5 = 0, 1 & \end{array}$$

Por lo que vamos al paso 12.

12. Como la columna $g^5_{\bullet 1} = g^5_{\bullet 7} = g^5_{\bullet 8} = g^5_{\bullet 10} = g^5_{\bullet 11}$, formamos nuevas cubiertas principales al PSC con matriz G_5: {5, 7}, {5, 8}, {5, 10}, {5, 11}.

Por lo tanto las cubiertas principales para el campo generador i = 5 son: {1, 5}, {5, 7}, {5, 8}, {5, 10}, {5, 11}.

13. Generamos los *edits* implícitos a partir de las cubiertas principales aplicando el Lema 2.1.

$$\{1,5\} \leftrightarrow e^*\left(5,\{e_1,e_5\}\right): \langle \{1\},\{0,1\},\{0\},\{0\},A_5,A_6\rangle = F$$
$$\{5,7\} \leftrightarrow e^*\left(5,\{e_5,e_7\}\right): \langle \varnothing,A_2,\{0\},\varnothing,A_5,A_6\rangle = F$$
$$\{5,8\} \leftrightarrow e^*\left(5,\{e_5,e_8\}\right): \langle \{1\},A_2,\{0\},\{0\},A_5,\{0,1\}\rangle = F$$

$$\{5,10\} \leftrightarrow e^{\bullet}\left(5,\{e_5,e_{10}\}\right):\langle\{1\},\{0,1\},A_3,\{0\},A_5,\{2,3\}\rangle = F$$
$$\{5,11\} \leftrightarrow e^{\bullet}\left(5,\{e_5,e_{11}\}\right):\langle\{1\},\{1\},A_3,\varnothing,A_5,\{2,3\}\rangle = F$$

De los cuales, los esencialmente nuevos no redundantes son $e^{\bullet}\left(5,\{e_1,e_5\}\right)$, $e^{\bullet}\left(5,\{e_5,e_8\}\right)$ y $e^{\bullet}\left(5,\{e_5,e_{10}\}\right)$ que corresponden, respectivamente, a e_{12}, e_{13} y e_{14} en la Tabla 4.1.

Consideremos el campo generador $i = 6$. Entonces el algoritmo es el siguiente:

1. Tenemos que:

$$G_6 = \begin{array}{c} \\ 0 \\ 1 \\ 2 \\ 3 \end{array}\begin{array}{c} \begin{array}{ccccccccc} e_2 & e_4 & e_6 & e_8 & e_9 & e_{10} & e_{11} & e_{13} & e_{14} \end{array} \\ \begin{pmatrix} 0 & 1 & 0 & 1 & 1 & 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 & 1 & 0 & 0 & 1 & 0 \\ 1 & 0 & 1 & 0 & 0 & 1 & 1 & 0 & 1 \\ 1 & 0 & 1 & 0 & 0 & 1 & 1 & 0 & 1 \end{pmatrix} \end{array}$$

Con PSC asociado:

$$\begin{array}{lccccccccccccc} \text{Minimizar} & x_2 & + & x_4 & + & x_6 & + & x_8 & + & x_9 & + & x_{10} & + x_{11} & + x_{13} & + x_{14} & \\ \text{Sujeto a} & & & x_4 & + & & & x_8 & + & x_9 & + & & & x_{13} & & \geq 1 \\ & x_2 & + & & & x_6 & + & & & & & x_{10} & + x_{11} & + & x_{14} & \geq 1 \\ & & & & & x_2, x_4, x_6, x_8, x_9, x_{10}, x_{11}, x_{13}, x_{14} = 0\ ,\ 1 & & & & & & & & & & \end{array}$$

En esta matriz observamos que la columna de e_2 es igual a la de e_6, e_{10}, e_{11}, e_{14} y la columna de e_4 es igual a la de e_8, e_9 y e_{13} por lo que removemos las columnas que corresponden a e_6, e_8, e_9, e_{10}, e_{11}, e_{13} y e_{14} obteniendo:

$$H_6 = \begin{array}{c} \\ 0 \\ 1 \\ 2 \\ 3 \end{array}\begin{array}{c} \begin{array}{cc} e_2 & e_4 \end{array} \\ \begin{pmatrix} 0 & 1 \\ 0 & 1 \\ 1 & 0 \\ 1 & 0 \end{pmatrix} \end{array}$$

2. Para cada pareja (u, v), $0 \leq u < v \leq 3$, es decir, para las parejas (0,1), (0,2), (0,3), (1,2), (1,3) y (2,3) verificamos si $h^6_{u\bullet} \geq h^6_{v\bullet}$. Como $h^6_{0\bullet} \geq h^6_{1\bullet}$ y $h^6_{2\bullet} \geq h^6_{3\bullet}$, se eliminan las filas $h^6_{0\bullet}$ y $h^6_{2\bullet}$ y por lo tanto:

$$B_6 = \begin{array}{c} \\ 1 \\ 3 \end{array}\begin{array}{c} \begin{array}{cc} e_2 & e_4 \end{array} \\ \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix} \end{array}$$

3. Observamos que las filas unitarias son: $I_2^3 = (0 \quad 1) = b_{1\bullet}^6$, $I_4^1 = (1 \quad 0) = b_{3\bullet}^6$.

4. Al eliminar de B_6 las dos filas y las dos columnas obtenemos la matriz reducida $B_{16} = \varnothing$.

5. Como $l = m_6 = 2$, el conjunto {2, 4} es la única cubierta principal para el PSC:

$$\begin{array}{lll} \text{Minimizar} & x_2 + x_4 & \\ \text{Sujeto a} & x_2 & \geq 1 \\ & x_4 & \geq 1 \\ & x_2, x_4 = 0, 1 & \end{array}$$

Por lo que vamos al paso 12.

12. Como la fila $g_{\bullet 2}^6 = g_{\bullet 6}^6 = g_{\bullet 10}^6 = g_{\bullet 11}^6 = g_{\bullet 14}^6$ y $g_{\bullet 4}^6 = g_{\bullet 8}^6 = g_{\bullet 9}^6 = g_{\bullet 13}^6$, formamos nuevas cubiertas principales al PSC con matriz G_6: {2, 8}, {2, 9}, {2, 13}, {4, 6}, {4, 10}, {4, 11}, {4, 14}, {6, 8}, {6, 9}, {6, 13}, {8, 10}, {8, 11}, {8, 14}, {9, 10}, {9, 11}, {9, 14}, {10, 13}, {11, 13}, {13, 14}.

Por lo tanto, las cubiertas principales para el campo generador i = 6 son: {2, 4}, {2, 8}, {2, 9}, {2, 13}, {4, 6}, {4, 10}, {4, 11}, {4, 14}, {6, 8}, {6, 9}, {6, 13}, {8, 10}, {8, 11}, {8, 14}, {9, 10}, {9, 11}, {9, 14}, {10, 13}, {11, 13}, {13, 14}.

13. Generamos los *edits* implícitos a partir de las cubiertas principales aplicando el Lema 2.1.

$$\{2,4\} \leftrightarrow e^*(6,\{e_2,e_4\}) : \langle \{1\},\{0,2\},\{1\},\{0,1\},A_5,A_6\rangle = F$$
$$\{2,8\} \leftrightarrow e^*(6,\{e_2,e_8\}) : \langle \{1\},A_2,\varnothing,\{0,1\},\{0,1\},A_6\rangle = F$$
$$\{2,9\} \leftrightarrow e^*(6,\{e_2,e_9\}) : \langle \varnothing,A_2,\{1\},\{1\},A_5,A_6\rangle = F$$
$$\{2,13\} \leftrightarrow e^*(6,\{e_2,e_{13}\}) : \langle \{1\},A_2,\varnothing,\{0\},A_5,A_6\rangle = F$$
$$\{4,6\} \leftrightarrow e^*(6,\{e_4,e_6\}) : \langle A_1,\{2\},\{1\},\{1\},A_5,A_6\rangle = F$$
$$\{6,8\} \leftrightarrow e^*(6,\{e_6,e_8\}) : \langle A_1,\{1,2\},\varnothing,\{1\},\{0,1\},A_6\rangle = F$$
$$\{6,9\} \leftrightarrow e^*(6,\{e_6,e_9\}) : \langle \{0\},\{1,2\},\{1\},\{1\},A_5,A_6\rangle = F$$
$$\{6,13\} \leftrightarrow e^*(6,\{e_6,e_{13}\}) : \langle \{1\},\{1,2\},\varnothing,\varnothing,A_5,A_6\rangle = F$$
$$\{8,10\} \leftrightarrow e^*(6,\{e_8,e_{10}\}) : \langle \{1\},\{0,1\},\{0\},\{0,1\},\{0,1\},A_6\rangle = F$$
$$\{8,11\} \leftrightarrow e^*(6,\{e_8,e_{11}\}) : \langle A_1,\{1\},\{0\},\{1\},\{0,1\},A_6\rangle = F$$
$$\{8,14\} \leftrightarrow e^*(6,\{e_8,e_{14}\}) : \langle \{1\},\{0,1\},\{0\},\{0\},\{0,1\},A_6\rangle = F$$

$$\{4,10\} \leftrightarrow e^*(6,\{e_4,e_{10}\}):\langle\{1\},\{0\},A_3,\{0,1\},\{0,1\},A_6\rangle = F$$
$$\{4,11\} \leftrightarrow e^*(6,\{e_4,e_{11}\}):\langle A_1,\varnothing,A_3,\{1\},\{0,1\},A_6\rangle = F$$
$$\{4,14\} \leftrightarrow e^*(6,\{e_4,e_{14}\}):\langle\{1\},\{0\},A_3,\{0\},A_5,A_6\rangle = F$$
$$\{9,10\} \leftrightarrow e^*(6,\{e_9,e_{10}\}):\langle\varnothing,\{0,1\},A_3,\{1\},\{0,1\},A_6\rangle = F$$
$$\{9,11\} \leftrightarrow e^*(6,\{e_9,e_{11}\}):\langle\{0\},\{1\},A_3,\{1\},\{0,1\},A_6\rangle = F$$
$$\{9,14\} \leftrightarrow e^*(6,\{e_9,e_{14}\}):\langle\varnothing,\{0,1\},A_3,\varnothing,A_5,A_6\rangle = F$$
$$\{10,13\} \leftrightarrow e^*(6,\{e_{10},e_{13}\}):\langle\{1\},\{0,1\},\{0\},\{0\},\{0,1\},A_6\rangle = F$$
$$\{11,13\} \leftrightarrow e^*(6,\{e_{11},e_{13}\}):\langle\{1\},\{1\},\{0\},\varnothing,\{0,1\},A_6\rangle = F$$
$$\{13,14\} \leftrightarrow e^*(6,\{e_{13},e_{14}\}):\langle\{1\},\{0,1\},\{0\},\{0\},A_5,A_6\rangle = F$$

De los cuales, los esencialmente nuevos no redundantes son $e^*(6,\{e_2,e_4\})$, $e^*(6,\{e_4,e_6\})$, $e^*(6,\{e_4,e_{10}\})$ y $e^*(6,\{e_4,e_{14}\})$ que corresponden, respectivamente, a e_{15}, e_{16}, e_{17} y e_{18} en la Tabla 4.1.

La siguiente tabla nos muestra los *edits* implícitos esencialmente nuevos no redundantes, su campo generador y sus *edits* contribuyentes.

Campo generador	***Edits* contribuyentes**	***Edit* implícito esencialmente nuevo no redundante**
1	e_2, e_3	$e_6:\langle A_1,\{1,2\},\{1\},\{1\},A_5,\{2,3\}\rangle = F$
2	e_1, e_3	$e_7:\langle\{0\},A_2,\{0\},\{1,2,3\},\{0,1\},A_6\rangle = F$
	e_1, e_4	$e_8:\langle A_1,A_2,\{0\},A_4,\{0,1\},\{0,1\}\rangle = F$
	e_3, e_4	$e_9:\langle\{0\},A_2,A_3,\{1,2,3\},A_5,\{0,1\}\rangle = F$
3	e_1, e_2	$e_{10}:\langle\{1\},\{0,1\},A_3,\{0,1\},\{0,1\},\{2,3\}\rangle = F$
	e_1, e_6	$e_{11}:\langle A_1,\{1\},A_3,\{1\},\{0,1\},\{2,3\}\rangle = F$
5	e_1, e_5	$e_{12}:\langle\{1\},\{0,1\},\{0\},\{0\},A_5,A_6\rangle = F$
	e_5, e_8	$e_{13}:\langle\{1\},A_2,\{0\},\{0\},A_5,\{0,1\}\rangle = F$
	e_5, e_{10}	$e_{14}:\langle\{1\},\{0,1\},A_3,\{0\},A_5,\{2,3\}\rangle = F$
6	e_2, e_4	$e_{15}:\langle\{1\},\{0,2\},\{1\},\{0,1\},A_5,A_6\rangle = F$
	e_4, e_6	$e_{16}:\langle A_1,\{2\},\{1\},\{1\},A_5,A_6\rangle = F$
	e_4, e_{10}	$e_{17}:\langle\{1\},\{0\},A_3,\{0,1\},\{0,1\},A_6\rangle = F$
	e_4, e_{14}	$e_{18}:\langle\{1\},\{0\},A_3,\{0\},A_5,A_6\rangle = F$

Podemos observar, a partir de los ejemplos 5.2, 6.2 y 7.6, que el Algoritmo BCC tiene un mejor desempeño que los Algoritmos GKL1 y WW, todos los cuales intentan obtener el conjunto completo de *edits*, al obtener dicho conjunto de manera más rápida gracias a la incorporación del PSC y a la reducción de la matriz G_i.

CAPÍTULO VIII

EXTENSIONES A LA METODOLOGÍA DE FELLEGI Y HOLT

En este capítulo presentamos extensiones a la teoría y a los aspectos computacionales de la metodología de Fellegi y Holt. Si los *edits* implícitos pueden ser generados antes de la validación, entonces la ***localización de errores*** (LE), i.e., encontrar el mínimo número de campos a imputar, puede ser bastante rápida. En algunas situaciones, no todos los *edits* implícitos pueden ser generados por el gran número de combinaciones de *edits* explícitos e implícitos previamente obtenidos a verificar ($>10^{30}$).

Las ideas presentadas en este capítulo pretenden determinar más rápidamente el número mínimo aproximado de campos a cambiar en situaciones donde no todos los *edits* implícitos pueden ser generados antes de la validación, por lo que proveeremos un algoritmo para la LE. Los algoritmos son mucho más rápidos que los métodos directos de programación entera para la LE, que no usan *edits* implícitos calculados antes. El incremento en la velocidad se debe a que los métodos directos de programación entera generan *edits* implícitos durante la LE, por lo que muchos de éstos son calculados repetidamente.

Existen tres métodos de LE. El primero y más lento es el que utiliza métodos directos de programación entera como el Branch and Bound. El método puede requerir alrededor de 10 minutos por registro. El segundo método emplea variantes del Algoritmo de Chernikova con cardinalidad restringida. El método es usado en el Sistema GEIS del Buró Canadiense de Estadística (en C, un segundo por registro), en el Sistema CherryPi del Buró Holandés

de Estadística (en Pascal, 2 segundos por registro) y en el Sistema AGGIES del Servicio Nacional de Estadística en Agricultura (SAS, más de un minuto por registro).

Debido a que los métodos son algo lentos, el Buró Holandés de Estadística desarrolló el sistema LEO que emplea variantes del método de eliminación de Fourier-Motzkin (en Pascal, al menos 10 registros por segundo). Ninguno de esos métodos o sistemas puede lidiar, sin embargo, con grandes encuestas (con millones de registros). Por ejemplo, con el Censo de Manufactura de Estados Unidos, el 4% de 2.5 millones de registros (100,000) son registros con errores. Esta gran cantidad de registros excede drásticamente la capacidad de los sistemas antes mencionados. No es posible validar manualmente 100,000 registros. Ya que muchos de estos registros están asociados a pequeñas compañías, parece razonable desarrollar un sistema que implemente la metodología de Fellegi y Holt que pueda validar e imputar todos los registros automáticamente. Sólo los registros asociados a las compañías más grandes serían revisados manualmente como un paso adicional en la validación.

En este capítulo se demuestra cómo a todos los registros se les puede realizar la LE cuando no todos los *edits* implícitos puedan ser generados antes. Si la mayoría de los *edits* implícitos son generados antes de la validación, entonces es de esperarse que a la mayoría de los registros se les puede realizar la LE. Los métodos de este capítulo proveen un medio de LE para una pequeña proporción de registros restantes que no puedan ser corregidos correctamente debido a un conjunto incompleto de *edits* implícitos. Los métodos son mucho más rápidos que aquellos basados en los Algoritmos de Chernikova o de eliminación de Fourier-Motzkin.

8.1 RESULTADOS TEÓRICOS.

Esta sección contiene teoría y explicaciones que tratan de hacer la comprensión de los algoritmos computacionales tan fácil como sea posible. Aún más, mostramos cómo un registro y^0 es rellenado. Por ***rellenar***, queremos decir cómo nuevos valores son imputados en los campos de y^0 para que ya no falle ningún *edit* explícito.

Daremos dos restricciones que pueden ser hechas, sin pérdida de generalidad, en términos de la teoría y de las aplicaciones prácticas en software.

1) Cada *edit* e_r tiene al menos dos campos entrantes.

Si un *edit* e_r tuviera un solo campo entrante, digamos j, entonces j tendría al menos un valor que siempre resultaría en un error, sin importar qué valores tomaran los otros campos. Se puede lidiar mejor con estos *edits* de un solo campo mediante pre-*edits*.

2) Para cada n, $A_n = \bigcup_r \{A_n^r \mid e_r \in \underline{e}_E\}$ donde $\underline{e}_E$ es el conjunto original de *edits* explícitos definidos por los analistas.

Si la unión fuera un subconjunto propio de A_n para alguna n, entonces cualquier registro y^0 con un componente y_n^0 en A_n pero no en la unión pasaría necesariamente todos los *edits*.

La segunda restricción significa que consideramos sólo valores de campos que entran en al menos un *edit* y que no hay valores de campos individuales que no entran en al menos un campo en un *edit*. En la práctica, estas restricciones podrían fácilmente ser revisadas vía rutinas combinatorias directas. Esto aliviaría una revisión tediosa por analistas. Las restricciones facilitan nuestro desarrollo teórico pero no afectan el desarrollo de software.

En este capítulo, estamos interesados en realizar la LE cuando el conjunto de *edits* $\underline{e}$ está incompleto. Dos enfoques podrían ser considerados. El primero es usar un método heurístico para determinar rápidamente campos adicionales que podrían ser cambiados. Del primer enfoque hay dos variantes. En la primera variante, ejemplificada por la metodología de imputación del vecino más cercano (MIVC), el registro donador produce un conjunto J^* que contiene al conjunto de campos que deben ser cambiados. En la segunda variante, identificamos un conjunto preliminar de campos J a cambiar (basados en un conjunto incompleto de *edits* $\underline{e}$). El conjunto preliminar es también extendido a un conjunto J^* que debe ser cambiado. En cada una de las variantes, un subconjunto es entonces identificado representando los campos a cambiar.

En el segundo enfoque, algunos *edits* implícitos adicionales son localizados en el curso de la LE. Hay dos variantes. En este capítulo, usamos un método que utiliza información obtenida durante el proceso de generación de *edits* (Capítulos VI y VII). El algoritmo GKL2 es usado para identificar todos los *edits* que fallan durante la LE.

El Algoritmo GKL2 proporciona una manera de encontrar todos los *edits* fallados adicionales para un registro y^0. Además, si tomamos cualquier campo entrante i_0 en un *edit* implícito $\hat{e}$, podemos, iterativamente, expandir la cubierta inicial J a $J_1 = J \cup \{i_0\}$. Al finalizar el proceso de iteración, se obtiene una cubierta principal de los *edits* fallados por el registro y^0. Como se observa en el Capítulo V, el número excesivo ($\prod_{j \in J^*} n_j$) de patrones del paso 2 del algoritmo, hace a este proceso computacionalmente irrastreable excepto en muy pequeñas situaciones. Nuestra alternativa al Algoritmo GKL2 aliviará muchos de los cálculos excesivos utilizando mucha más información disponible del proceso de generación de un conjunto incompleto de *edits*.

Sea el conjunto de *edits* explícitos e implícitos $\underline{e}$ incompleto. Si y es un registro que falla un *edit* implícito que no está en $\underline{e}$, entonces indicamos qué puede ir mal a medida que el registro es rellenado. Después, proporcionamos una caracterización detallada de cómo ocurren condiciones contradictorias cuando una cubierta J^* de los *edits* fallados por un registro y^0 no cubre un campo en un *edit* implícito fallado que no está en $\underline{e}$. Mostramos cómo determinar rápidamente campos adicionales J_1^* tales que al cambiar los valores en

los campos de $J^* \cup J_1^*$ obtengamos un registro y' que satisfaga todos los *edits* explícitos. Finalmente, proporcionamos el teorema principal que muestra, para cualquier conjunto Y de registros, cómo generar *edits* implícitos faltantes.

Para presentar los resultados teóricos de este capítulo necesitamos plantear y explicar un conjunto de lemas técnicos que son importantes porque dan la clave de cómo los cálculos computacionales pueden ser reducidos.

Los Lemas 8.1 y 8.2 nos proporcionarán las ideas claves para las siguientes secciones. Sea J una cubierta principal del conjunto de *edits* en el conjunto incompleto $\underline{e}$ que fallan el registro y^0. Intentamos mostrar que J puede ser extendida a una cubierta principal J^* del conjunto de todos los *edits* fallados por y^0. Parte de nuestro desarrollo involucrará demostrar cómo *edits* explícitos que no eran fallados originalmente por el registro y^0 pueden fallar al ser cambiados los campos en la cubierta J.

Lema 8.1:

Sea J una cubierta principal de un conjunto incompleto de *edits* fallados $\underline{e}_F(y^0)$ para el registro y^0. Entonces, J puede ser expandida a una cubierta principal $J^* = J \cup \{f_1,...,f_n\}$ que es una cubierta principal para todos los *edits* fallados por y^0.

Demostración

Como $\underline{e}$ es un conjunto incompleto de *edits*, no todas las restricciones necesarias para resolver el PSC dado por (4.4)–(4.6) estarán incluidas en él. El Algoritmo GKL2 nos da todos los *edits* implícitos faltantes para completar $\underline{e}_F(y^0)$. Supongamos que tenemos m nuevos *edits* que impondrán las correspondientes m restricciones al PSC dado por (4.4)–(4.6). Si J cubría las anteriores restricciones, las nuevas restricciones requerirán n nuevos campos $f_1,...f_n$ a agregar a J, donde $n \in \{0,1,...,m\}$. ■

Sin embargo, hay que hacer notar que no necesariamente esta cubierta principal encontrada por el lema será la de peso mínimo pues las nuevas restricciones podrían hacer que un conjunto distinto de campos que no incluyen a los de J fuera el de peso mínimo.

Ejemplo 8.1:

Retomando el Ejemplo 5.3 para el Algoritmo GKL2, supongamos que de los 18 *edits* del ejemplo, únicamente nos falta el *edit* e_{12}. Así, para el registro $y^0 = (1 \quad 0 \quad 0 \quad 0 \quad 1 \quad 0)$, $J = \{5,6\}$ con costo 6. Sin embargo, al aplicar el Algoritmo

se obtendrá el *edit* e_{12} que al imponer una nueva restricción al PSC dado por (4.4)–(4.6) dará como cubierta principal de peso mínimo $J = \{2,3,5\}$ con costo 6.

Nuestra meta es encontrar métodos computacionalmente rastreables para expandir un conjunto incompleto de *edits* y encontrar rápidamente cubiertas principales de todos los *edits* fallados por todos los registros $y^0 \in Y$, donde Y son todos los registros que fallan algún *edit*.

Lema 8.2:

Sean $\underline{e}_C$ un conjunto completo de *edits* y y^0 un registro que falla algún *edit*. Entonces $\underline{e}_C$ da información acerca de *edits* explícitos no fallados por y^0 que podrían fallar al cambiar ciertos campos en y^0.

Demostración

Primero debemos observar que un registro que falla algún *edit* implícito necesariamente falla uno explícito, ya que si un registro pasa todos los *edits* explícitos, pasará todos los *edits* en $\underline{e}_C$.

Al fallar nuestro registro un *edit* explícito lo hará por entrar en los campos entrantes del *edit*. Al querer imputar debemos decidir cuál o cuáles de estos campos entrantes vamos a cambiar. Es ahí donde los *edits* implícitos nos dan información ya que, si observamos que un *edit* implícito tuvo como uno de sus *edits* contribuyentes a nuestro *edit* fallado, entonces el campo generador no será un campo cuyo cambio sea suficiente para pasar todos los *edits* en $\underline{e}_C$ ya que el complemento de los valores que toma un *edit* contribuyente en un campo entran en el otro *edit*.

Así conviene más cambiar un campo que entre tanto en el *edit* fallado como en todos los *edits* implícitos que lo involucren de alguna manera. ■

Cualquier cubierta principal J^* de los campos entrantes del conjunto completo $\underline{e}_C$ de *edits* explícitos e implícitos que fallan puede llevar a un registro y' que satisface todos los *edits*. El registro y' difiere del original sólo en los valores de los campos en la cubierta J^*. Cualquier cubierta principal J^* de los *edits* explícitos e implícitos fallados se dice que es una solución de la LE. Ésta satisface las condiciones (4.7) y (4.8).

Sea S^* el conjunto de todos los campos que no están en J^*. Comenzamos con valores fijos en cada campo de S^*. Sean $f_1,...,f_n$ los campos en J^*. El registro y puede ser rellenado al encontrar sucesivamente valores para cada campo f_k en J^* tales que el registro con los

valores fijos para todos los campos en S^* y los campos $f_1', ..., f_n'$ satisfaga todos los *edits* explícitos e implícitos que tienen campos entrantes en $S^* \cup \{f_1', ..., f_n'\}$. Con un conjunto completo de *edits* explícitos e implícitos $\underline{e}_C$, es directo encontrar una cubierta J^* y rellenar un registro.

Decimos que un valor para un campo está ***asignado*** si el valor está fijo. Si un registro tiene valores asignados en los campos de $S^* \cup \{f_1, ..., f_k\}$, necesitamos encontrar un valor para el campo f_{k+1}. Consideramos todos los *edits* explícitos e implícitos $e_1, ..., e_n$ que tienen un campo entrante f_{k+1} y el resto de sus campos en $S^* \cup \{f_1, ..., f_k\}$. Por la Sección 3.2, el complemento de las intersecciones del conjunto de valores para el campo entrante f_{k+1} en los *edits* $e_1, ..., e_n$ no debe ser vacío. Podemos escoger cualquier valor para f_{k+1} que esté en ese conjunto.

Un ***edit potencialmente fallado*** (explícito o implícito) es aquel que puede fallar en algunas situaciones cuando un campo entrante en un *edit* explícito o implícito inicialmente fallado es cambiado. Obsérvese que un registro potencialmente fallado no falla inicialmente.

Supongamos que un campo en un registro es cambiado y un *edit* potencialmente fallado falla ahora. Tome un segundo campo entrante del *edit* potencialmente fallado original y cambie el valor del registro de forma que el nuevo valor no esté en el conjunto de valores del segundo campo entrante. Entonces, el registro necesariamente pasará ambos *edits*.

Ejemplo 8.2:

> Consideremos una situación simple. Sea $y \in Y$ un registro que tiene cuatro campos. Supongamos que cada campo tiene dos valores entrantes. Sean los *edits* implícitos e_1 con campos entrantes f_1 y f_2 y e_2 con campos entrantes f_2 y f_3. Supongamos que i_1 es un *edit* implícito generado de e_1 y e_2 usando el campo f_2 como generador, que el registro y falla e_2 e i_1, que el valor en el campo entrante f_1 para el registro y está entre los valores del campo entrante para el *edit* e_1 y que sólo tenemos los *edits* explícitos e_1 y e_2. El *edit* i_1 no está disponible. Sea $J^* = \{f_2\}$ la cubierta del *edit* fallado e_2 y $S^* = \{f_1, f_3, f_4\}$. Cambie el valor en el campo f_2 para obtener así un registro y_1. Entonces e_1 falla. Cambiamos ahora el campo f_1 para obtener el registro y_2. Este registro pasa e_1 y e_2.

El ejemplo anterior puede ser extendido a cualquier situación donde haya un *edit* implícito faltante. Esto es, si un *edit* implícito falta en una cubierta, entonces un campo en una cubierta del conjunto incompleto de *edits* explícitos e implícitos existente puede ser cambiado de manera que haga que un *edit* potencialmente fallado e_k falle. Si uno de los campos en el *edit* implícito faltante que coincida con otro campo en el *edit* potencialmente

fallado i_2 es también cambiado, entonces el *edit* e_k no fallará. Observemos que si un conjunto de valores es asignado (basados en el conjunto inicial S^* y el subconjunto de valores en los campos de J^* que han sido cambiados) y si un *edit* implícito falla, entonces un *edit* explícito con campos entrantes en el conjunto de valores asignados necesariamente fallará.

Sea J^* una cubierta de un conjunto $\underline{e}$ de *edits* explícitos e implícitos de primer nivel fallados. Sea $\underline{e}_k$ el conjunto de *edits* explícitos no fallados que tienen exactamente un campo entrante f_k que coincide con un campo en J^* y que falla cuando el valor del campo f_k en J^* es cambiado. Necesariamente el valor ubicado en J^* debe estar en el complemento de las intersecciones del conjunto de valores para el campo entrante f_k en los *edits* explícitos e implícitos de primer nivel originalmente fallados. Sea f_1 otro campo entrante en $\underline{e}_k$. Sea I^* el conjunto de los campos complementarios en todos estos *edits* explícitos no fallados que fallan después que un campo de J^* es cambiado. Sea S_1^* el complemento de los campos en $J^* \bigcup I^*$. Entonces S_1^* es un conjunto propio de S^*. S_1^* puede no ser un conjunto maximal de campos asignados. Un conjunto propio de campos asignados es ***maximal*** si ningún superconjunto de él es propio.

Supongamos que tenemos un conjunto incompleto de *edits* explícitos e implícitos existe. En lo que resta de este capítulo suponemos que, como mínimo, el conjunto de *edits* implícitos debe incluir todos los *edits* implícitos de primer nivel. Hay muchas formas diferentes en las que la LE podría ser realizada en el programa de validación principal. Primero, si el conjunto incompleto de *edits* se supone que está casi completo, entonces podría ser mejor tomar una cubierta J^* de los *edits* fallados en el conjunto incompleto y equivocarse al rellenar el registro. Podría ser suficiente encontrar campos adicionales que se necesitaran agregar al conjunto de campos en J^*. Segundo, podría ser mejor buscar inmediatamente campos I^* para agregar a la cubierta J^* antes de realizar la LE. Tercero, si a $\underline{e}$ le faltan un número moderado de *edits* implícitos, entonces podría ser mejor generar *edits* implícitos adicionales basados en el conjunto de *edits* fallados existente. La generación sería un híbrido eficiente del Algoritmo GKL2 que tenga como objetivo un solo campo cada vez. Pruebas apropiadas que encuentren *edits* implícitos adicionales pueden ser realizadas antes de correr el programa de validación principal.

Examinemos ahora las situaciones más simples. Supongamos que cada campo sólo toma dos valores, que cada *edit* contiene sólo dos campos entrantes y que no existen blancos por pase. Las extensiones para cuando cada campo toma más de dos valores son directas. Las extensiones a más de dos campos entrantes en un *edit* son tediosas. Las extensiones para cuando hay blancos por pase no son triviales (ver Sección 8.3).

Supongamos que el escenario de validación involucra registros que tienen n campos. Los únicos *edits* implícitos que consideramos son *edits* implícitos no redundantes.

Lema 8.3 :

Suponga que no hay blancos por pase asociados al conjunto de *edits*. Entonces todo *edit* implícito no redundante de nivel n es generado por un *edit* implícito de nivel $n-1$ y un *edit* explícito.

Demostración

Si el lema no fuera cierto, entonces existiría un conjunto de *edits* implícitos no redundantes generados por dos *edits* implícitos como contribuyentes. Usando el orden dado por el recorrido del BCC2, sea e el primer *edit* implícito generado por dos *edits* implícitos (e_5 y e_6) en el campo f_3 y que no puede ser generado por un *edit* implícito y uno explícito. Entonces e_5 y e_6 son generados, cada uno, por un par de *edits*, uno de los cuales, al menos, es explícito. Supongamos que e_5 es generado por los *edits* e_1 y e_2 en el campo f_1, donde e_1 es explícito y que e_6 es generado por los *edits* e_3 y e_4 en el campo f_2, donde e_3 es explícito. Consideramos ahora varios casos:

1) e entra en los campos f_1 y f_2.

Por construcción, el campo f_3 puede entrar o no en los *edits* e_2 y e_4.

a) Si el campo f_3 entra en los *edits* e_2 y e_4, entonces $e(f_3,\{e_2,e_4\})$ es un *edit* no redundante generado antes que e. Por lo tanto, e_2 ó e_4 es un *edit* explícito. Pero $e(f_3,\{e_2,e_4\})=e$, de aquí que e es un *edit* generado por un *edit* implícito de nivel superior y un *edit* explícito con lo que llegamos a una contradicción.
b) Si el campo f_3 no entra en el *edit* e_2 y entra en el e_4, entonces e es un *edit* redundante pues es cubierto por e_2, lo cual es una contradicción.
c) Por simetría se obtiene una contradicción para el caso en el que el campo f_3 no entra en el *edit* e_4 y entra en el e_2 y para el caso en el que el campo f_3 no entra en ninguno de los *edits*.

2) e no entra en el campo f_1 y entra en el campo f_2.

Por construcción, el campo f_3 puede entrar o no en el *edit* e_4.

a) Si el campo f_3 entra en el *edit* e_4, entonces $e(f_3,\{e_4,e_5\})$ es un *edit* no redundante generado antes que e. Por lo tanto, e_4 ó e_5 es un *edit* explícito. Pero $e(f_3,\{e_4,e_5\})=e$, de aquí que e es un *edit* generado

por un *edit* implícito de nivel superior y un *edit* explícito con lo que llegamos a una contradicción.

b) Si el campo f_3 no entra en el *edit* e_4, entonces e es un *edit* redundante pues es cubierto por e_4, lo cual es una contradicción.

3) e no entra en el campo f_2 y entra en el campo f_1.

Por construcción, el campo f_3 puede entrar o no en el *edit* e_2.

a) Si el campo f_3 entra en el *edit* e_2, entonces $e(f_3,\{e_2,e_6\})$ es un *edit* no redundante generado antes que e. Por lo tanto, e_2 ó e_6 es un *edit* explícito. Pero $e(f_3,\{e_2,e_6\})=e$, de aquí que e es un *edit* generado por un *edit* implícito de nivel superior y un *edit* explícito con lo que llegamos a una contradicción.

b) Si el campo f_3 no entra en el *edit* e_2, entonces e es un *edit* redundante pues es cubierto por e_2, lo cual es una contradicción.

4) e no entra en los campos f_1 y f_2.

Por construcción el campo f_3 debe entrar en al menos uno de los *edits* e_1 y e_2 y en al menos uno de los *edits* e_3 y e_4.

a) Si el campo f_3 entra en los *edits* e_1 y e_3 y no entra en los *edits* e_2 y e_4, entonces e_6 domina a e_3. Así, e_6 es ahora un *edit* explícito y e es un *edit* generado por un *edit* implícito de nivel superior y un *edit* explícito, con lo que llegamos a una contradicción.

b) Si el campo f_3 entra en los *edits* e_1 y e_4 y no entra en los *edits* e_2 y e_3, entonces e_5 domina a e_1. Así, e_5 es ahora un *edit* explícito y e es un *edit* generado por un *edit* implícito de nivel superior y un *edit* explícito, con lo que llegamos a una contradicción.

c) Si el campo f_3 entra en los *edits* e_2 y e_3 y no entra en los *edits* e_1 y e_4, entonces generamos los *edits* $e_7=e(f_3,\{e_3,e_5\})$ y $e(f_2,\{e_4,e_7\})$. Así, $e(f_2,\{e_4,e_7\})$ es un *edit* no redundante generado antes que e, entonces e_4 es un *edit* explícito. Pero $e(f_2,\{e_4,e_7\})$ domina a e, por lo que lo sustituye y $e(f_2,\{e_4,e_7\})$ es un *edit* generado por un *edit* implícito y uno explícito.

d) Si el campo f_3 entra en los *edits* e_2 y e_4 y no entra en los *edits* e_1 y e_3, entonces generamos el *edit* $e_7=e(f_1,\{e_1,e_8\})$, donde e_8 es uno

de los *edits* generadores de e_2. Generamos ahora el *edit* $e(f_3,\{e_6,e_7\})$ que es no redundante y generado antes que e. Entonces, $e(f_3,\{e_6,e_7\})$ es un *edit* generado por un *edit* implícito y uno explícito. Pero $e(f_3,\{e_6,e_7\})$ domina a e, por lo que lo sustituye.

e) Si el campo f_3 entra en los *edits* e_2, e_3 y e_4 y no entra en el *edit* e_1, entonces e_6 domina a e_3. Así, e_6 es ahora un *edit* explícito y e es un *edit* generado por un *edit* implícito de nivel superior y un *edit* explícito, con lo que llegamos a una contradicción.

f) Si el campo f_3 entra en los *edits* e_1, e_3 y e_4 y no entra en el *edit* e_2, entonces e_5 domina a e_1. Así, e_5 es ahora un *edit* explícito y e es un *edit* generado por un *edit* implícito de nivel superior y un *edit* explícito, con lo que llegamos a una contradicción.

g) Si el campo f_3 entra en los *edits* e_1, e_2 y e_4 y no entra en el *edit* e_3, entonces e_5 domina a e_1. Así, e_5 es ahora un *edit* explícito y e es un *edit* generado por un *edit* implícito de nivel superior y un *edit* explícito, con lo que llegamos a una contradicción.

h) Si el campo f_3 entra en los *edits* e_1, e_2 y e_3 y no entra en el *edit* e_4, entonces e_5 domina a e_1 y e_6 domina a e_3. Así, e_5 y e_6 son ahora *edits* explícitos y e es un *edit* generado por dos *edits* explícitos, con lo que llegamos a una contradicción.

i) Si el campo f_3 entra en los *edits* e_1, e_2, e_3 y e_4, entonces e_5 domina a e_1 y e_6 domina a e_3. Así, e_5 y e_6 son ahora *edits* explícitos y e es un *edit* generado por dos *edits* explícitos, con lo que llegamos a una contradicción. ■

Lema 8.4:

Sea e un *edit* implícito en el nivel n que es generado por el *edit* e_1 de nivel $n-1$ y otro *edit* e_2 usando el campo f_1. Sea y un registro que falla e. Si y falla el *edit* e_1 y el valor en el campo f_1 es cambiado, entonces el registro y' falla e_2. Si y falla e_2 y el valor en el campo f_1 es cambiado, entonces el registro y' falla el *edit* e_1.

Demostración

Tenemos los *edits* contribuyentes:

$$e_1 : A_1^1 \times A_2^1 \times \cdots \times A_{f_1}^1 \times \cdots \times A_n^1 = F$$
$$e_2 : A_1^2 \times A_2^2 \times \cdots \times A_{f_1}^2 \times \cdots \times A_n^2 = F$$

Entonces el *edit* implícito está dado por:

$$e : A_1^1 \cap A_1^2 \times A_2^1 \cap A_2^2 \times \cdots \times A_{f_1}^1 \cup A_{f_1}^2 \times \cdots A_n^1 \cap A_n^2 = A_1^1 \cap A_1^2 \times A_2^1 \cap A_2^2 \times \cdots \times A_{f_1} \times \cdots A_n^1 \cap A_n^2 = F$$

Como el registro y falla el *edit* e entonces

$$y \in A_1^1 \cap A_1^2 \times A_2^1 \cap A_2^2 \times \cdots \times A_{f_1} \times \cdots A_n^1 \cap A_n^2 .$$

Si y falla el *edit* e_1 entonces

$$y \in (A_1^1 \cap A_1^2 \times A_2^1 \cap A_2^2 \times \cdots \times A_{f_1} \times \cdots A_n^1 \cap A_n^2) \cap (A_1^1 \times A_2^1 \times \cdots \times A_{f_1}^1 \times \cdots \times A_n^1)$$
$$= A_1^1 \cap A_1^2 \times A_2^1 \cap A_2^2 \times \cdots \times A_{f_1}^1 \times \cdots A_n^1 \cap A_n^2 .$$

Entonces al cambiar el valor de y en el campo f_1 tenemos que

$$y' \in A_1^1 \cap A_1^2 \times A_2^1 \cap A_2^2 \times \cdots \times A_{f_1}^{1,c} \times \cdots A_n^1 \cap A_n^2 = A_1^1 \cap A_1^2 \times A_2^1 \cap A_2^2 \times \cdots \times A_{f_1}^2 \times \cdots A_n^1 \cap A_n^2 = E_2$$

Por lo tanto, y' falla el *edit* e_2. Análogamente, si y falla e_2 y el valor en el campo f_1 es cambiado, entonces el registro y' falla el *edit* e_1.■

Sea e un *edit* en el nivel n. Sea y un registro que falla e. Decimos que $C = \{e_1, e_2, ..., e_n\}$ es una ***y-cadena*** que lleva al *edit* e si e_1 es un *edit* explícito (es decir, de nivel 0), $e = e_n$ y el registro y falla todos los *edits* en C. No es necesario que las y-cadenas sean únicas.

Lema 8.5:

Suponga que $\underline{e}$ es un conjunto incompleto de *edits*. Sean e_0 un *edit* implícito faltante en $\underline{e}$ y y un registro que falla el *edit* e_0. Entonces existe una y-cadena que lleva a e_0.

Demostración

Supongamos que e_0 sería generado en el nivel n si lo hubiera sido y que $n \geq 1$. Por el Lema 8.3, e_0 es generado por un *edit* implícito e_0^{n-1} en el nivel $n-1$ y por el *edit* explícito e_{n-1}^0. Debido a que y falla e_0 debe fallar necesariamente e_0^{n-1} ó

e^0_{n-1}. Establezca que e_2 (igual a e^{n-1}_0 o a e^0_{n-1}) es fallado. Si e_2 es igual a e^0_{n-1}, entonces hemos construido una *y*-cadena que va de e^n_0 a e^0_{n-1}. Para cada *j*, con $0 < j < n-1$, suponga que la *y*-cadena no ha terminado aún y sea e^j_0 igual a e_2 el *edit* implícito en el nivel *j* que falla. Entonces existe el *edit* implícito e^{j-1}_0 en el nivel $j-1$ y el *edit* explícito e^0_{j-1} que son usados para generar e^j_0. Sea e'_2 igual a e^{j-1}_0 o a e^0_{j-1} dependiendo de cuál falle. Si e'_2 es igual a e^0_{j-1} entonces la *y*-cadena termina, si no continuamos la inducción hasta que $j-1=0$. ■

Sea *y* un registro que falla un *edit* e_0 de nivel *n*. Sea $C=\{e_1,e_2,...,e_n\}$ una *y*-cadena que lleva a e_0. Supongamos que $D=\{f_1,f_2,...,f_{n-1}\}$ son *edits* implícitos que satisfacen la siguiente propiedad: para $j>1$, e_j tiene *edits* contribuyentes e_{j-1} y f_j. Aplicando inductivamente los Lemas 8.3-8.5, tenemos que el registro *y* no puede fallar ninguno de los *edits* en *D*. Sin embargo, si el valor en el campo generador es cambiado, entonces es posible que algún *edit* explícito falle. Si $\underline{e}$ está incompleto, entonces algunos de los *edits* en la *y*-cadena *C* no estarán en $\underline{e}$.

Lema 8.6:

Sean $\underline{e}$ un conjunto incompleto de *edits*, *i* un *edit* implícito que no está en $\underline{e}$, *y* un registro que falla el *edit* implícito *i*, $\underline{e}_F(y)$ el conjunto de *edits* implícitos y explícitos en $\underline{e}$ que son fallados por *y*, $J=\{f_1,f_2,...,f_n\}$ cualquier cubierta principal de $\underline{e}_F(y)$ que no incluya ningún campo entrante de *i* e *y'* el registro obtenido de *y* al cambiar cada uno de los valores en *J*. Entonces *y'* falla al menos un *edit* explícito e_0.

Demostración

El registro *y'* necesariamente falla el *edit* implícito *i* debido a que ningún campo entrante en *i* fue cambiado. Por el Lema 8.5 hay una *y*-cadena que lleva de algún *edit* explícito *e* a un *edit* implícito *i*.

Por lo tanto, el registro *y'* necesariamente falla el *edit* explícito e_0 al principio de la *y*-cadena. Necesariamente existe una *y*-cadena *C* que va de e_0 a *i*. No todos los *edits* en la *y*-cadena *C* necesitan estar en $\underline{e}$. ■

Lema 8.7:

Sean $\underline{e}$ un conjunto incompleto de *edits*, e_1 en el nivel más bajo, y un registro que falla el *edit* e_1, f_1 y f_2 campos entrantes en e_1 e y' el registro obtenido de y al cambiar f_1 ó f_2. Suponga que y' falla un *edit* explícito e_2 y que y no falla. Entonces es posible generar un *edit* implícito e_3 con *edits* contribuyentes e_1 y e_2 tales que e_3 sea fallado por y.

Demostración

Supongamos que y' difiere de y en el campo f_1. Tenga e_2 campos entrantes f_3 y f_4. Entonces, necesariamente el campo f_3 o el f_4 debe ser igual al campo f_1 ya que si el campo f_1 no entrara en e_2, entonces y e y' no diferirían en ningún campo entrante de e_2, por lo que y fallaría e_2, lo cual es una contradicción. Supongamos que $f_1 = f_4$. Generamos el *edit* e_3 con el campo f_1 como generador y campos entrantes f_2 y f_3. Por lo anterior tenemos que:

$$e_1 : A_1 \times \cdots \times A_{f_1}^1 \times \cdots \times A_{f_2}^1 \times \cdots \times A_{f_3} \times \cdots \times A_n = F$$
$$e_2 : A_1 \times \cdots \times A_{f_1}^2 \times \cdots \times A_{f_2} \times \cdots \times A_{f_3}^2 \times \cdots \times A_n = F$$
$$\Downarrow$$
$$e_3 : A_1 \times \cdots \times A_{f_1} \times \cdots \times A_{f_2}^1 \times \cdots \times A_{f_3}^2 \times \cdots \times A_n = F$$

Claramente y entra en todos los campos ya que y' entra en $A_{f_3}^2$. Por lo tanto, y falla e_3.■

Lema 8.8:

Sean $\underline{e}$ un conjunto incompleto de *edits*, e_1 en el nivel más bajo, y un registro que falla el *edit* e_1 y f_1 y f_2 campos entrantes en e_1. Suponga que el *edit* e_1 es generado por un *edit* implícito e_3 y uno explícito e_4 en el campo f_3, que el registro y falla e_3 pero no e_4 y que f_1 entra en e_4. Sea y' el registro obtenido de y al cambiar el campo f_3. Entonces el registro y' puede o no fallar el *edit* e_4. Suponga que el registro y' falla el *edit* explícito e_4. Sea y_1 el registro que difiere de y' al cambiar el campo f_1. Entonces el registro y_1 no falla e_4, e_3 ni e_1.

Demostración

Como en el *edit* e_1 el campo f_3 fue el generador entonces está completo. Como y e y' difieren únicamente en el campo f_3 y fallan e_3 y e_4, respectivamente, éstos necesariamente deben tener en f_3 valores ajenos y complementarios. Anexando las hipótesis del lema tenemos que los *edits* son de la siguiente forma:

$$e_3 : A_1 \times \cdots \times A_{f_1}^{(1)} \times \cdots \times A_{f_2}^{(1)} \times \cdots \times A_{f_3}^{3} \times \cdots \times A_n = F$$
$$e_4 : A_1 \times \cdots \times A_{f_1}^{1} \times \cdots \times A_{f_2}^{(1)} \times \cdots \times (A_{f_3}^{3})^c \times \cdots \times A_n = F$$
$$\Downarrow$$
$$e_1 : A_1 \times \cdots \times A_{f_1}^{1} \times \cdots \times A_{f_2}^{1} \times \cdots \times A_{f_3} \times \cdots \times A_n = F$$

donde (1) significa que el campo puede estar completo o ser el correspondiente a e_1. Hay que señalar que en el caso del campo f_2, para obtener un subconjunto propio en e_1 al menos uno de los campos f_2 en los *edits* contribuyentes debe ser entrante.

Claramente observamos que al cambiar f_1 en y' obtenemos un registro y_1 que pasa los tres *edits*. ■

Lema 8.9:

Sean $\underline{e}$ un conjunto incompleto de *edits* e y un registro que falla algunos de los *edits* en $\underline{e}$. Entonces, es posible expandir $\underline{e}$ a un conjunto de *edits* $\underline{e}'$ que contiene todos los *edits* que son fallados por el registro y.

Demostración

Sea $\underline{e}_0$ el subconjunto de $\underline{e}$ de los *edits* en el nivel más bajo. Definimos conjuntos de *edits* en el nivel más bajo que el registro y falla como sigue:

Sea $\underline{e}_0(y)$ el subconjunto de $\underline{e}_0$ de los *edits* que fallan el registro y. Sea $e_1 \in \underline{e}_0(y)$. Por el Lema 8.6, si los campos entrantes de e_1 son cambiados, entonces no fallará ningún *edit* o un *edit* explícito que no fallaba inicialmente fallará. En el primer caso definimos $\underline{e}_1(y) = \underline{e}_0(y) \setminus \{e_1\}$. En el otro caso, definimos $\underline{e}_1(y) = \underline{e}_0(y) \cup \{e_2\} \setminus \{e_1\}$, donde e_2 es el *edit* implícito sucesor de e_1 (i.e., e_2 tiene como *edit* contribuyente de nivel inmediato superior a e_1) que es fallado por y.

Escogemos $e_3 \in \underline{e}_1(y)$. Si e_3 no tiene sucesor, sea $\underline{e}_2(y) = \underline{e}_1(y) \setminus \{e_3\}$. Si e_3 tiene sucesor e_4 que es fallado por y sea $\underline{e}_2(y) = \underline{e}_1(y) \cup \{e_4\} \setminus \{e_3\}$.

En general, para $j > 2$, sea e_{2j+1} un *edit* en $\underline{e}_j(y)$. Si e_{2j+1} no tiene sucesor, definimos $\underline{e}_{j+1}(y) = \underline{e}_j(y) \setminus \{e_{2j+1}\}$. Si e_{2j+1} tiene sucesor e_{2j+2}, definimos $\underline{e}_{j+1}(y) = \underline{e}_j(y) \cup \{e_{2j+2}\} \setminus \{e_{2j+1}\}$. Ya que el registro y sólo puede fallar un número finito de *edits*, debe existir un valor de j para el cuál $\underline{e}_{j+1}(y) = \varnothing$. Sea $\underline{e}' = \underline{e} \cup \{\bigcup_j e_{2j+2}\}$. ■

Durante el proceso de generación de *edits* implícitos, es fácil rastrear en qué campo el *edit* implícito fue generado, qué *edits* contribuyeron y en qué nivel el *edit* implícito fue generado. Para cada *edit* implícito es posible encontrar una y-cadena (no única) que nos lleva de regreso a un *edit* explícito en el nivel cero. Ya que estamos interesados en los *edits* implícitos que puedan faltar del conjunto de *edits* que hemos generado, necesitamos sólo considerar *edits* implícitos que estén al final de una y-cadena. En otras palabras, si $\underline{e}$ es el conjunto existente de *edits* explícitos e implícitos, entonces sólo consideramos *edits* implícitos $\underline{e}_f$ en $\underline{e}$ que están en el nivel más bajo de las y-cadenas.

Si tenemos un conjunto incompleto $\underline{e}$ de *edits* explícitos e implícitos, entonces el Lema 8.7 nos da una manera muy rápida de determinar campos adicionales que llevarían a una solución de LE. Si generamos y almacenamos los *edits* implícitos adicionales que son obtenidos al aplicar repetidamente el Lema 8.7, entonces tenemos todos los *edits* implícitos que un registro y falla. Podemos aplicar nuevamente la rutina principal de LE para obtener una solución más eficiente que sería obtenida al agregar campos f_E a los campos de la cubierta del conjunto incompleto $\underline{e}$. Podemos concebir a éste como un enfoque heurístico rápido para encontrar una solución de LE.

Supongamos que un conjunto incompleto $\underline{e}$ contiene la mayoría de los *edits* implícitos. Es muy rápido expandir $\underline{e}$ a $\underline{e}'$ con *edits* implícitos encontrados vía el Lema 8.9. Para un conjunto dado de registros Y, si $\underline{e}$ es expandido a $\underline{e}'$, entonces el conjunto $\underline{e}'$ contendría necesariamente todos los *edits* implícitos fallados por Y. La generalización del Lema 8.9 a situaciones en las que un campo puede tomar más de dos valores es directa. La generalización en situaciones cuando están presentes blancos por pase (en la forma de la encuesta y en el conjunto de *edits*) no es directa.

Estamos ahora en posición de establecer el teorema principal.

Teorema 8.1:

Sean $\underline{e}$ un conjunto incompleto de *edits*, Y un conjunto de registros que fallan *edits* en $\underline{e}$, $y \in Y$ un registro que falle al menos un *edit* implícito que no está en $\underline{e}$, $\underline{e}_F(y)$ el conjunto de *edits* implícitos en $\underline{e}$ fallados por el registro y y $J = \{f_1, f_2, ..., f_n\}$ el conjunto de campos que son cubierta principal de $\underline{e}_F(y)$. Entonces es posible encontrar rápidamente un conjunto de campos $J_y = \{f_{n+1}, ..., f_q\}$ tal que $J \cup J_y$ es una solución de LE para y. Aún más, es posible encontrar los *edits* implícitos $\underline{e}_{my}$ que son fallados por y y no están en $\underline{e}_F(y)$.

Demostración

El Lema 8.1 nos garantiza que se puede extender la cubierta J de modo que ésta cubra todos los *edits* fallados por y. Los Lemas 8.2-8.8 nos permiten detectar de forma más rápida los elementos a anexar a la cubierta para obtener un registro que pase todos los *edits*. El Lema 8.6 nos garantiza el último enunciado del teorema.■

Corolario 8.1:

Sean $\underline{e}$ un conjunto incompleto de *edits* e Y un conjunto de registros que fallan *edits* en $\underline{e}$. Entonces $\underline{e}$ puede ser extendido a un conjunto de *edits* $\underline{e}'$ que contienen todos los *edits* implícitos fallados por los registros en Y.

Demostración

Aplicando el Teorema 8.1 a cada registro de Y, llegamos al resultado deseado.■

8.2 ALGORITMOS CW.

Para este algoritmo se considera que además de los *edits* explícitos (dados por los especialistas) se tienen al menos los *edits* implícitos de primer nivel.

En la situación cuando no hay blancos por pase, tenemos el siguiente algoritmo.

Algoritmo CW1[*]*:*

1. Sea $\underline{e}_F(y^0)$ el conjunto de *edits* fallados por el registro y^0. Resuelva el PSC (4.8) y denote la solución por x^*.

[*]Con base en el primer algoritmo dado en William E. Winkler y Bor-Chung Chen (2002).

2. Sea $J = \{j \mid x_j^* = 1\}$. Fije los valores y_j^0 con $j \notin J$ en. Para cada $j \in J$, sea $E_j^c = \{(\bigcap A_j^i)^c \mid j \in J, e_i \in \underline{e}_F(y^0)\}$ el complemento de la intersección de los campos j en los *edits* fallados por y^0 y con estos valores forme las posibles combinaciones de registros.

3. Si cada nuevo registro lleva a un *edit* explícito nuevamente fallado e_k, entonces genere un nuevo *edit* implícito i_k que falle el registro y^0 usando como *edits* contribuyentes los *edits* explícitos que fueron fallados nuevamente y como campos generadores los elementos de J. Si para todo nuevo registro no existen dichos e_k, entonces x^* es una solución de (4.7); si no, sea $\underline{e}_I = \{i_k\}$ el conjunto de nuevos *edits* implícitos. Sea $\underline{e}_F(y^0) = \underline{e}_F(y^0) \cup \underline{e}_I$. Vaya al paso 1.

Ejemplo 8.3:

Retomemos el Ejemplo 4.1.

Tenemos 6 campos:

$$A_1 = \{0,1\} \quad n_1 = 2 \qquad A_4 = \{0,1,2,3\} \quad n_4 = 4$$
$$A_2 = \{0,1,2\} \quad n_2 = 3 \qquad A_5 = \{0,1,2\} \quad n_5 = 3$$
$$A_3 = \{0,1\} \quad n_3 = 2 \qquad A_6 = \{0,1,2,3\} \quad n_6 = 4$$

Los *edits* explícitos son:

$$e_1 : \langle A_1, \{0,1\}, \{0\}, A_4, \{0,1\}, A_6 \rangle = F,$$
$$e_2 : \langle \{1\}, A_2, \{1\}, \{0,1\}, A_5, \{2,3\} \rangle = F,$$
$$e_3 : \langle \{0\}, \{1,2\}, A_3, \{1,2,3\}, A_5, A_6 \rangle = F,$$
$$e_4 : \langle A_1, \{0,2\}, A_3, A_4, A_5, \{0,1\} \rangle = F,$$
$$e_5 : \langle \{1\}, A_2, A_3, \{0\}, \{1,2\}, A_6 \rangle = F.$$

Y los *edits* implícitos de primer nivel son:

$$e_6 : \langle A_1, \{1,2\}, \{1\}, \{1\}, A_5, \{2,3\} \rangle = F$$
$$e_7 : \langle \{0\}, A_2, \{0\}, \{1,2,3\}, \{0,1\}, A_6 \rangle = F$$
$$e_8 : \langle A_1, A_2, \{0\}, A_4, \{0,1\}, \{0,1\} \rangle = F$$
$$e_9 : \langle \{0\}, A_2, A_3, \{1,2,3\}, A_5, \{0,1\} \rangle = F$$
$$e_{10} : \langle \{1\}, \{0,1\}, A_3, \{0,1\}, \{0,1\}, \{2,3\} \rangle = F$$

$$e_{12} : \langle \{1\}, \{0,1\}, \{0\}, \{0\}, A_5, A_6 \rangle = F$$
$$e_{15} : \langle \{1\}, \{0,2\}, \{1\}, \{0,1\}, A_5, A_6 \rangle = F$$

Además $C = (5 \quad 3 \quad 1 \quad 4 \quad 2 \quad 4)$, $y^0 = (1 \quad 0 \quad 0 \quad 0 \quad 1 \quad 0)$. Entonces el algoritmo es el siguiente:

1. Observemos que $\underline{e}_F(y^0) = \{e_1, e_4, e_5, e_8, e_{12}\}$. Entonces el PSC (4.8) es:

$$\begin{array}{lrcrcrcrcrcl}
\text{Minimizar} & 5x_1 & + & 3x_2 & + & x_3 & + & 4x_4 & + & 2x_5 & + & 4x_6 & \\
\text{Sujeto a} & & & x_2 & + & x_3 & + & & & x_5 & & & \geq 1 \\
 & & & x_2 & + & & & & & & & x_6 & \geq 1 \\
 & x_1 & + & & & & & x_4 & + & x_5 & & & \geq 1 \\
 & & & & & x_3 & + & & & x_5 & + & x_6 & \geq 1 \\
 & x_1 & + & x_2 & + & x_3 & + & x_4 & & & & & \geq 1 \\
 & & & & & x_1, \ldots, x_6 = 0, 1 & & & & & & &
\end{array}$$

La cubierta solución es $x^* = (0 \quad 1 \quad 0 \quad 0 \quad 1 \quad 0)$ pues x_2 y x_5 cubren todas las restricciones. El costo es 5.

2. Tenemos que $J = \{2,5\}$ y los E_j^c son:

$$E_2^c = (A_2^1 \cap A_2^4 \cap A_2^5 \cap A_2^8 \cap A_2^{12})^c = (\{0,1\} \cap \{0,2\} \cap A_2 \cap A_2 \cap \{0,1\})^c = (\{0\})^c = \{1,2\}$$
$$E_5^c = (A_5^1 \cap A_5^4 \cap A_5^5 \cap A_5^8 \cap A_5^{12})^c = (\{0,1\} \cap A_5 \cap \{1,2\} \cap \{0,1\} \cap A_5)^c = (\{1\})^c = \{0,2\}$$

Tenemos 4 posibles registros a verificar dados en la siguiente tabla junto con los *edits* que fallan.

Posible registro	***Edit*(*s*) fallado(s) por el registro**
$(1 \quad 1 \quad 0 \quad 0 \quad 0 \quad 0)$	e_1, e_8, e_{12}
$(1 \quad 1 \quad 0 \quad 0 \quad 2 \quad 0)$	e_5, e_{12}
$(1 \quad 2 \quad 0 \quad 0 \quad 0 \quad 0)$	e_4, e_8
$(1 \quad 2 \quad 0 \quad 0 \quad 2 \quad 0)$	e_4, e_5

3. Los *edits* explícitos nuevamente fallados son e_1, e_4 y e_5. Entonces, generamos un *edit* implícito usando a los anteriores *edits* como contribuyentes y como campos generadores los campos 2 y 5. Por lo tanto, el *edit* generados es:

$$\hat{e}:\langle\{1\},A_2,\{0\},\{0\},A_5,\{0,1\}\rangle = F$$

que corresponde a e_{13} en la Tabla 4.1. Entonces $\underline{e}_F(y^0) = \{e_1, e_4, e_5, e_8, e_{12}, e_{13}\}$.

1'. Al PSC (4.8) del paso 1 agregamos la restricción $x_1 + x_3 + x_4 + x_6 \geq 1$ dada por e_{13}. Entonces una cubierta solución es $x^* = (0 \quad 1 \quad 1 \quad 0 \quad 1 \quad 0)$ con costo 6.

2'. Tenemos que $J^* = \{2,3,5\}$ y los E_j^c son:

$$E_2^c = (A_2^1 \cap A_2^4 \cap A_2^5 \cap A_2^8 \cap A_2^{12} \cap A_2^{13})^c = (\{0,1\} \cap \{0,2\} \cap A_2 \cap A_2 \cap \{0,1\} \cap A_2)^c = (\{0\})^c = \{1,2\}$$
$$E_3^c = (A_3^1 \cap A_3^4 \cap A_3^5 \cap A_3^8 \cap A_3^{12} \cap A_3^{13})^c = (\{0\} \cap A_3 \cap A_3 \cap \{0\} \cap \{0\} \cap \{0\})^c = (\{0\})^c = \{1\}$$
$$E_5^c = (A_5^1 \cap A_5^4 \cap A_5^5 \cap A_5^8 \cap A_5^{12} \cap A_5^{13})^c = (\{0,1\} \cap A_5 \cap \{1,2\} \cap \{0,1\} \cap A_5 \cap A_5)^c = (\{1\})^c = \{0,2\}$$

Tenemos 4 posibles registros a verificar dados en la siguiente tabla junto con los *edits* que fallan.

Posible registro	***Edit*(s) fallado(s) por el registro**
$(1 \quad 1 \quad 1 \quad 0 \quad 0 \quad 0)$	
$(1 \quad 1 \quad 1 \quad 0 \quad 2 \quad 0)$	e_5
$(1 \quad 2 \quad 1 \quad 0 \quad 0 \quad 0)$	e_4, e_{15}
$(1 \quad 2 \quad 1 \quad 0 \quad 2 \quad 0)$	e_4, e_5

De aquí observamos que el registro $(1 \quad 1 \quad 1 \quad 0 \quad 0 \quad 0)$ no falla ningún *edit*. Por lo tanto, la solución a (4.7) es $x^* = (0 \quad 1 \quad 1 \quad 0 \quad 1 \quad 0)$ con costo 6.

Un algoritmo alternativo mucho más rápido es:

*Algoritmo CW2**:*

1. Sea $\underline{e}_F(y^0)$ el conjunto de *edits* fallados por el registro y^0. Resuelva el PSC (4.8) y denote la solución por x^*.
2. Sea $J = \{j \mid x_j^* = 1\}$.

*Con base en el segundo algoritmo dado en William E. Winkler y Bor-Chung Chen (2002).

3. Fije los valores de y_j^0 con $j \notin J$. Para cada $j \in J$, sea $E_j^c = \{(\bigcap A_j^i)^c \mid j \in J, e_i \in \underline{e}_F(y^0)\}$ el complemento de la intersección de los campos j en los *edits* fallados por y^0. Con estos valores forme las posibles combinaciones de registros. Si cada nuevo registro lleva a un *edit* explícito nuevamente fallado $e_{k(j)}$, escójase de entre estos *edits* aquél que tenga el mínimo número ponderado de campos entrantes que no estén en J. Sea D dicho conjunto de campos. Si, para cada nuevo registro, no existe dicho e_k, vaya al paso 4; en otro caso, sea $J = J \cup D$. Vaya al paso 2.

4. Encuentre un subconjunto de J que lleve a una solución de (4.7).

Ejemplo 8.4:

Retomemos el Ejemplo 8.3. Entonces el algoritmo es el siguiente:

1. Observemos que $\underline{e}_F(y^0) = \{e_1, e_4, e_5, e_8, e_{12}\}$. Entonces el PSC (4.8) es:

$$
\begin{array}{lrcrcrcrcrcl}
\text{Minimizar} & 5x_1 & + & 3x_2 & + & x_3 & + & 4x_4 & + & 2x_5 & + & 4x_6 & \\
\text{Sujeto a} & & & x_2 & + & x_3 & + & & & x_5 & & & \geq 1 \\
 & & & x_2 & + & & & & & & & x_6 & \geq 1 \\
 & x_1 & + & & & & & x_4 & + & x_5 & & & \geq 1 \\
 & & & x_3 & + & & & & & x_5 & + & x_6 & \geq 1 \\
 & x_1 & + & x_2 & + & x_3 & + & x_4 & & & & & \geq 1 \\
 & & & & & x_1, \ldots, x_6 = 0, 1 & & & & & & &
\end{array}
$$

La cubierta solución es $x^* = (0 \quad 1 \quad 0 \quad 0 \quad 1 \quad 0)$ pues x_2 y x_5 cubren todas las restricciones. El costo es 5.

2. Tenemos que $J = \{2,5\}$.

3. Los E_j^c son:

$$E_2^c = (A_2^1 \cap A_2^4 \cap A_2^5 \cap A_2^8 \cap A_2^{12})^c = (\{0,1\} \cap \{0,2\} \cap A_2 \cap A_2 \cap \{0,1\})^c = (\{0\})^c = \{1,2\}$$
$$E_5^c = (A_5^1 \cap A_5^4 \cap A_5^5 \cap A_5^8 \cap A_5^{12})^c = (\{0,1\} \cap A_5 \cap \{1,2\} \cap \{0,1\} \cap A_5)^c = (\{1\})^c = \{0,2\}$$

Tenemos 4 posibles registros a verificar dados en la siguiente tabla junto con los *edits* que fallan.

Posible registro	*Edit*(*s*) fallado(s) por el registro
(1 1 0 0 0 0)	e_1, e_8, e_{12}
(1 1 0 0 2 0)	e_5, e_{12}
(1 2 0 0 0 0)	e_4, e_8
(1 2 0 0 2 0)	e_4, e_5

Los *edits* explícitos nuevamente fallados son e_1, e_4 y e_5. El campo entrante de e_1 distinto de 2 y 5 es el 3 con peso 1, de e_4 es el 6 con peso 4 y de e_5 son el 1 y el 4 con peso 9. Por lo que tomamos el campo 3 y $D = \{3\}$.

2'. Tenemos que $J^* = \{2,3,5\}$.

3'. Los E_j^c son:

$$E_2^c = (A_2^1 \cap A_2^4 \cap A_2^5 \cap A_2^8 \cap A_2^{12} \cap A_2^{13})^c = (\{0,1\} \cap \{0,2\} \cap A_2 \cap A_2 \cap \{0,1\} \cap A_2)^c = (\{0\})^c = \{1,2\}$$
$$E_3^c = (A_3^1 \cap A_3^4 \cap A_3^5 \cap A_3^8 \cap A_3^{12} \cap A_3^{13})^c = (\{0\} \cap A_3 \cap A_3 \cap \{0\} \cap \{0\} \cap \{0\})^c = (\{0\})^c = \{1\}$$
$$E_5^c = (A_5^1 \cap A_5^4 \cap A_5^5 \cap A_5^8 \cap A_5^{12} \cap A_5^{13})^c = (\{0,1\} \cap A_5 \cap \{1,2\} \cap \{0,1\} \cap A_5 \cap A_5)^c = (\{1\})^c = \{0,2\}$$

Tenemos 4 posibles registros a verificar dados en la siguiente tabla junto con los *edits* que fallan.

Posible registro	*Edit*(*s*) fallado(s) por el registro
(1 1 1 0 0 0)	
(1 1 1 0 2 0)	e_5
(1 2 1 0 0 0)	e_4, e_{15}
(1 2 1 0 2 0)	e_4, e_5

De aquí observamos que el registro (1 1 1 0 0 0) no falla ningún *edit*.

4. Por lo tanto, la solución a (4.7) es $x^* = (0\ 1\ 1\ 0\ 1\ 0)$ con costo 6.

El Algoritmo CW2 es mucho más rápido que el CW1 ya que los nuevos *edits* implícitos no necesitan ser calculados en etapas intermedias del algoritmo. El Algoritmo CW2 generalmente no llevará a una solución mínima de (4.7). El Algoritmo CW1 puede llevar a una solución mínima de (4.7) si un Algoritmo de Branch and Bound o uno similarmente apropiado es aplicado a (4.8) con el nuevo conjunto de *edits* implícitos que fallen el registro

y^0. El Algoritmo CW1 es mucho más rápido que el Algoritmo GKL2 porque no requiere enumerar todos los $\prod_{j \in J^*} n_j$ posibles registros asociados con la cubierta J y encontrar todos los *edits* existentes que los fallan.

8.3 SITUACIÓN CON BLANCOS POR PASE.

Cuando están presentes blancos por pase deseamos encontrar rápidamente los *edits* explícitos fallados al cambiarse los valores en los campos en J.

Para dar una idea intuitiva, describimos una situación típica en la que ocurren blancos por pase. Siempre supondremos que no hay blancos por pase dentro de blancos por pase. En otras palabras, nuestros blancos por pase son de primer nivel. Los blancos por pase de mayor nivel son extremadamente raros en la realidad. Para una encuesta de fuerza de trabajo, una pregunta puede solicitar la edad del encuestado. Si la edad es menor o igual a 12, entonces hay un salto a la sección de preguntas sobre educación. Si la edad es mayor a 12, entonces hay un salto a una sección de preguntas sobre fuerza de trabajo. Hay una suposición explícita de que los individuos con 12 o menos años de edad no estarán trabajando y que estarán en la escuela. Los individuos con más de 12 años que no contestan las preguntas sobre fuerza de trabajo se supone generalmente que están en la escuela. Equivalentemente, si un individuo contesta las preguntas sobre fuerza de trabajo, se supone que tienen más de 12 años.

Si no existen blancos por pase, entonces cada *edit* implícito en el nivel n puede ser generado por un *edit* implícito de nivel $n-1$ y un conjunto de *edits* explícitos. Si existen blancos por pase de primer nivel, entonces existen *edits* implícitos en el nivel n que sólo pueden ser generados por un *edit* implícito de nivel $n-1$, al menos un *edit* implícito de primer nivel y, posiblemente, *edits* explícitos adicionales. En el siguiente lema, suponemos que cada campo toma sólo dos valores. La extensión a más de dos valores es directa.

Lema 8.10:

Sean $y \in Y$ un registro que falla *edits* en un conjunto incompleto $\underline{e}$, e_1 un *edit* fallado en el nivel más bajo del árbol generador de *edits* existente, e_2 un *edit* implícito faltante que falla y que no será cubierto por ninguna cubierta principal J de los *edits* fallados en $\underline{e}$ y que sólo puede ser generado por e_1 y un *edit* implícito e_3 de primer nivel en un campo generador f_0. Fije una cubierta principal J que contiene al campo f_0. Si f_0 es cambiado, entonces existe un *edit* explícito e_4 que falla.

Demostración

Sea e_3 generado por e_{3a} y e_{3b} en algún campo f_1. Si f_0 es cambiado, entonces el *edit* e_3 falla. Dado que los campos entrantes de e_2 no son cubiertos por J, los campos entrantes de e_3 que son complementarios a f_0 tampoco son cubiertos.

Como e_3 falla, entonces e_{3a} ó e_{3b} falla. Supongamos que falla e_{3a}. Entonces f_0 entra e_{3a} que es el *edit* explícito e_4 buscado. Si un campo entrante complementario f_2 en e_{3a} también está en J, entonces cuando f_2 es cambiado e_{3a} ya no fallará. Si f_2 no está en J, entonces podemos expandir J con el campo f_2. Al cambiarse los valores en los campos J, un *edit* explícito eventualmente fallará.■

8.4 LA METODOLOGÍA DE IMPUTACIÓN DEL VECINO MÁS CERCANO (MIVC).

Muchos sistemas de imputación con donadores y de mínimo cambio están basados en la metodología de imputación de Fellegi y Holt (1976). Por ejemplo, CANEDIT y GEIS en Buró Canadiense de Estadística y DISCRETE y SPEER en el Buró del Censo de Estados Unidos están basados en la metodología de imputación de Fellegi y Holt.

Bankier (1991) introdujo un método exitoso de imputación *hot-deck* de donadores que ha sido usado en los Censos de Canadá de 1996 y 2001 y será usado para el de 2006. Al igual que con otros sistemas de imputación con donantes, el método requiere tener una gran cantidad de donantes de alta calidad.

Antes de describir la metodología de imputación del vecino más cercano (MIVC), describiremos cómo trabajaría un sistema de validación que implementara la metodología de Fellegi y Holt y que usara imputación *hot-deck*. El sistema de validación determinaría el mínimo número de campos a cambiar. Se desarrollarían reglas para seleccionar los donadores del conjunto de registros que satisfacen todos los *edits*. Si hay donadores apropiados, entonces los campos imputados de éstos mantendrán las distribuciones univariadas de los encuestados.

Dos dificultades están asociadas con sistemas que usan imputación *hot-deck*. La primera es que las reglas no sean las adecuadas. Esto ha sido indicado como un problema en el Censo Decenal de Estados Unidos de 1990, el Censo de Canadá de 1991 y el Censo de Inglaterra de 1991. La segunda es que puede no haber suficientes donadores apropiados. Frecuentemente el segundo problema no es tan serio en un censo como en una encuesta.

La MIVC fue usada en el Censo Canadiense de 1996 para realizar la validación e imputación para las variables Edad, Sexo, Estado civil y Relación con el jefe de familia. La

MIVC permite la imputación *hot-deck* simultánea de variables cualitativas y cuantitativas para grandes problemas de validación e imputación.

El siguiente es un ejemplo de validación e imputación de tres registros de individuos pertenecientes a un mismo hogar.

Ejemplo 8.5:

Considere los siguientes registros de un hogar con tres habitantes:

Relación con el jefe de familia	Estado civil	Edad
Jefe	Casado	38
Cónyuge	Casado	35
Madre	-	41

Tenemos una respuesta en blanco para el estado civil de la madre y la edad de ésta es inconsistente con la edad de su hijo (jefe). Los datos de un hogar que pasó los *edits* (donador) se usan para imputar este registro obteniendo el registro:

Relación con el jefe de familia	Estado civil	Edad
Jefe	Casado	38
Cónyuge	Casado	35
Madre	Viudo	59

Varios subconjuntos de las variables son imputados para determinar cuál es la imputación óptima para un registro que falló algún *edit*. Cada uno de estos subconjuntos, cuando se impute, será llamado una ***acción de imputación***.

La metodología de Fellegi y Holt primero determina el número mínimo de variables a imputar y después realiza la imputación, posiblemente buscando donadores. La MIVC, por el contrario, primero busca donadores y después determina el número mínimo de variables a imputar dados los donadores. Cambiar el orden de estas operaciones permite a la MIVC resolver problemas de validación e imputación más grandes y complejos. Sin embargo, la MIVC requiere de donadores para poder realizar la imputación.

En adelante, los algoritmos para aplicar la MIVC de forma eficiente a nivel de computadora se ilustrarán usando el Ejemplo 8.5.

8.5 CARACTERÍSTICAS DE LA MIVC.

Los objetivos de una metodología de imputación *hot-deck* deberían ser los siguientes:

a) El registro imputado deberá parecerse al registro fallado. Esto se logra, dado que se cuente con donadores, imputando el número mínimo de variables. La suposición subyacente (que no siempre es cierta en la práctica) es que el encuestado tiene más posibilidad de cometer uno o dos errores que de cometer muchos.

b) Los datos imputados del registro deberán proceder, si es posible, de un solo donador. Además, el registro imputado deberá parecerse a ese donador único. El alcanzar estos objetivos tenderá a asegurar que la combinación de respuestas imputadas y no imputadas en un registro sea consistente.

c) Las acciones de imputación igualmente buenas, basadas en donadores disponibles, deberán tener oportunidades similares de ser escogidas para evitar inflar falsamente el tamaño de grupos pequeños pero importantes en la población (e.g., personas con más de 100 años).

Estos objetivos los alcanza la MIVC al identificar primero como donadores potenciales aquellos registros que pasaron los *edits* y que son tan similares como sea posible al registro fallado. Con esto se quiere decir que los dos registros deberán coincidir en tantas variables cualitativas como sea posible y tener pequeñas diferencias entre las variables cuantitativas. Los registros con estas características serán llamados ***vecinos más cercanos***.

Para una pareja específica (registro fallado, vecino más cercano), los únicos candidatos para imputación son, por supuesto, aquellas variables que no coincidan. Una acción de imputación que pase los *edits* será llamada ***factible***. Una de estas acciones de imputación factibles que impute el menor número (o casi el menor número) de variables posible (que será llamada una ***acción de imputación de cambio casi mínimo*** o AICCM) es aleatoriamente seleccionada. Así, el registro imputado será tan similar como sea posible al registro fallado, así como al donador.

Estas AICCMs pueden ser identificadas eficientemente para cada vecino más cercano considerado como donador para un registro fallado de la siguiente forma:

a) Sólo los *edits* que alguna de las acciones de imputación posibles pueda fallar son conservados para cada pareja (registro fallado, vecino más cercano). Esto resulta en menos *edits* necesarios para evaluar las acciones de imputación.

b) Se consideran primero las variables que sea más posible que necesiten ser imputadas. Así, las respuestas en blanco e inválidas son imputadas primero seguidas por las variables que entren los *edits* que el registro falla (dado que al menos una de éstas debe ser imputada) y finalmente las otras variables.

c) Cuando se generan las acciones de imputación para una pareja (registro fallado, vecino más cercano), sólo aquellas que son cercanas a la óptima (es decir, son AICCMs) o esencialmente nuevas (i.e., ningún subconjunto de variables imputadas pasaría los *edits*) son evaluadas para factibilidad. Las acciones de imputación que no son esencialmente nuevas son descartadas porque una o más variables son

innecesariamente imputadas. Esto viola el principio de hacer el mínimo número de cambios posible a los datos.

8.6 APLICACIÓN DEL ALGORITMO MIVC A UN EJEMPLO.

El registro fallado en el Ejemplo 8.5 será usado para ilustrar la MIVC. Como en el Ejemplo 8.5, cada hogar se compone de tres miembros, así que cada hogar comprende tres registros con campos dados en la siguiente tabla con sus respectivos posibles valores.

Registro 1	Relación con el jefe de familia 1	{Jefe}
	Estado civil 1	{Soltero, Casado, Divorciado, Viudo, Separado}
	Edad 1	{0, ..., 120}
Registro 2	Relación con el jefe de familia 2	{Cónyuge, Madre, Padre, Hijo, Hija, Abuela, Abuelo, Nieto, Nieta, Suegro, Suegra, etc.}
	Estado civil 2	{Soltero, Casado, Divorciado, Viudo, Separado}
	Edad 2	{0, ..., 120}
Registro 3	Relación con el jefe de familia 3	{Cónyuge, Madre, Padre, Hijo, Hija, Abuela, Abuelo, Nieto, Nieta, Suegro, Suegra, etc.}
	Estado civil 3	{Soltero, Casado, Divorciado, Viudo, Separado}
	Edad 3	{0, ..., 120}

Los *edits* se especificarán mediante tablas de decisión binarias (TDBs) como se ilustra en la Tabla 8.1. Una TDB puede ser descrita como una matriz de 0's y 1's donde cada fila corresponde a una proposición (e.g., Relación con el jefe de familia 3 = Madre) y cada columna corresponde a un *edit*. Un 1 indica que la proposición entra en el *edit* mientras que un 0 indica lo contrario (e.g., el *edit* e_1 nos dice que si la tercera persona en el hogar es la madre del jefe de familia, la diferencia entre sus edades no debe ser menor a 20 años).

Proposición	***Edit***				
	e_1	e_2	e_3	e_4	e_5
Relación con el jefe de familia 3 = Madre	1	1	0	0	0
Edad 3 – Edad 1 < 20	1	0	0	0	0
Edad 3 < 35	0	1	0	0	0
Relación con el jefe de familia 3 = Abuela	0	0	1	1	0
Edad 3 – Edad 1 < 30	0	0	1	0	0
Edad 3 < 45	0	0	0	1	0
Relación con el jefe de familia 2 = Cónyuge	0	0	0	0	1
Relación con el jefe de familia 3 = Suegra	0	0	0	0	1
Edad 3 – Edad 2 < 20	0	0	0	0	1

Tabla 8.1. *TDB para el Ejemplo A.1.*

De aquí que el primer registro del Ejemplo 8.5 falla e_1, pues la diferencia entre las edades es de 3 años.

Se realiza una búsqueda entre los registros pasados para identificar los vecinos más cercanos al registro fallado del Ejemplo 8.5. Se da preferencia a aquellos registros que sean geográficamente cercanos. Uno de éstos es:

Relación con el jefe de familia	Estado civil	Edad
Jefe	Casado	**36**
Cónyuge	Casado	**37**
Suegra	**Viudo**	**59**

Las respuestas que no coinciden con el registro fallado están resaltadas. Asignando una medida de distancia en la que la distancia entre dos datos cuantitativos distintos siempre sea 1 y la distancia entre dos edades distintas sea de 0.05 por año, tenemos que este registro dista del fallado $2 + 2(0.05) + 2(0.05) + 18(0.05) = 3.1$.

Usando este vecino más cercano la respuesta en blanco (Estado civil 3) es imputada con el valor "Viudo". El registro resultante aún falla e_1. Si hubiera pasado los *edits*, nos habríamos detenido dado que ninguna otra imputación habría sido esencialmente nueva.

Para hacer más eficiente la imputación eliminamos los *edits* en los que los valores del registro fallado no estén involucrados, así como las proposiciones que sólo se vean involucradas en estos *edits*. Por ejemplo, el tercer miembro en el registro fallado del Ejemplo 8.5 tiene 41 años y 59 en el vecino más cercano. Entonces e_2 se elimina junto con la tercera proposición. Similarmente, ninguno de los registros tiene a una abuela como miembro del hogar; así e_3 y e_4 junto con las proposiciones 4, 5 y 6 son eliminadas; además eliminamos la proposición 7 pues el valor en el registro fallado (Cónyuge) coincide con el del vecino más cercano. Entonces nos quedamos únicamente con e_1 y e_5 y las cuatro proposiciones restantes, como se muestra en la Tabla 8.2.

Proposición	*Edit*	
	e_1	e_5
Relación con el jefe de familia 3 = Madre	1	0
Edad 3 – Edad 1 < 20	1	0
Relación con el jefe de familia 3 = Suegra	0	1
Edad 3 – Edad 2 < 20	0	1

Tabla 8.2. Edits *restantes después de la simplificación.*

Tenemos en estos *edits* cuatro variables involucradas (Relación con el jefe de familia 3, Edad 3, Edad 1, Edad 2), por lo que tenemos $2^4 - 1 = 15$ posibles acciones de imputación

(se resta uno pues la acción de imputación $(0\ \ 0\ \ 0\ \ 0)$ no tiene sentido) dadas en la Tabla 8.3, donde 1 representa la imputación de esa variable y 0 lo contrario.

Relación con el jefe de familia 3	Edad 3	Edad 1	Edad 2
1	0	0	0
0	**1**	**0**	**0**
0	0	1	0
0	0	0	1
1	1	0	0
1	0	1	0
1	0	0	1
0	1	1	0
0	**1**	**0**	**1**
0	0	1	1
1	1	1	0
1	1	0	1
1	0	1	1
0	1	1	1
1	1	1	1

Tabla 8.3. *Acciones de imputación considerando las variables Relación con el jefe de familia 3, Edad 3, 1 y 2.*

Para obtener un registro que pase el primer *edit* observamos que éste involucra las variables: Relación con el jefe de familia 3, Edad 3 y Edad 1, por lo que tenemos $2^3 - 1 = 7$ posibles acciones de imputación, a saber:

Relación con el jefe de familia 3	Edad 3	Edad 1
1	0	0
0	**1**	**0**
0	0	1
1	1	0
1	0	1
0	1	1
1	1	1

Tabla 8.4. *Acciones de imputación considerando las variables Relación con el jefe de familia 3, Edad 3 y 1.*

Dado que se imputa primero la variable cuya distancia del registro original al vecino más cercano es la menor, imputaremos primero Edad 1 seguido por Edad 3 y después por

Relación con el jefe de familia 3. Sólo algunas de estas acciones de imputación son generadas en la práctica como se demostrará ahora. Después de evaluar las dos primeras acciones de imputación, observamos que $(1\ \ 0\ \ 0)$ falla los *edits*, mientras que $(0\ \ 1\ \ 0)$ pasa los *edits*. Como resultado, las acciones de imputación $(1\ \ 0\ \ 1),(1\ \ 1\ \ 0)$ y $(1\ \ 1\ \ 1)$ no son generadas o evaluadas porque no serán esencialmente nuevas. Las dos acciones de imputación restantes $(0\ \ 0\ \ 1)$ y $(0\ \ 1\ \ 1)$ se observa que no pasan el segundo *edit* de la Tabla 8.2. Por lo que nos quedamos únicamente con la acción de imputación $(0\ \ 1\ \ 0)$, resaltada en la Tabla 8.4. Para esta acción de imputación corresponden las dos acciones de imputación resaltadas en la Tabla 8.3 que involucran o no la imputación de Edad 2. De estas dos, observamos que únicamente la acción de imputación $(0\ \ 1\ \ 0\ \ 0)$ pasa todos los *edits*.

8.7 CONSIDERACIONES ADICIONALES.

En la práctica, otras dos verificaciones son hechas para eliminar acciones de imputación adicionales:

- Si una acción de imputación no es una AICCM, es eliminada antes de aplicar los *edits* y las acciones de imputación adicionales de ésta no son generadas.

- Si una acción de imputación falla un *edit* y todas las variables en ese *edit* han sido ya consideradas para imputación, la acción de imputación será eliminada (junto con el *edit*) dado que la imputación de variables adicionales no permitirán que la acción de imputación resultante pase ese *edit*.

Este proceso de primero eliminar *edits* y proposiciones (y por lo tanto variables) y después generar y evaluar sólo un subconjunto de acciones de imputación, produce un algoritmo muy eficiente de imputación con cambio mínimo para grandes problemas.

El proceso de identificar acciones de imputación es repetido con otros vecinos más cercanos. Sea D_{fa} la distancia de la acción de imputación al registro fallado (i.e., una medida de cuántas variables son imputadas). Sea D_{ap} la distancia de la acción de imputación al vecino más cercano usado (i.e., una medida de certidumbre). Las n acciones de imputación con la D_{fap} más pequeña son retenidas, donde:

$$D_{fpa} = \alpha D_{fa} + (1-\alpha) D_{ap}$$

El parámetro α (el cual puede caer en el rango (0.05, 1]) da mayor o menor importancia a la imputación del mínimo número de variables. Entonces, una de esas n acciones de imputación es aleatoriamente seleccionada para ser la acción de imputación actual usada para el registro fallado.

La eficiencia computacional del algoritmo MIVC, a medida que el número de integrantes de un hogar aumenta, es ilustrada en la Tabla 8.5. Estos tiempos de CPU están estandarizados en términos del tiempo tomado para procesar un hogar unipersonal. Así, el desempeñar la validación y la imputación en un hogar de tres personas es 2.3 veces más costoso que desempeñar la validación y la imputación en un hogar unipersonal. El número de *edits* crece rápidamente a medida que crece el número de personas en el hogar, ya que los *edits* entre personas tienen que ser generados para todas las posibles parejas de personas en un hogar. Así, hay 307 *edits* para un hogar de tres personas y 2,435 para un hogar de seis personas. Mientras que hay ocho veces más *edits* para un hogar de seis personas que para uno de tres, los costos computacionales crecen sólo por un factor de 5. Los costos computacionales para hogares con siete y ocho personas son similares a los de hogares con seis personas debido a la reducción en el número de donadores para grandes registros.

Número de individuos en el hogar	Número de *edits*	Tiempo estandarizado en términos del tiempo para un hogar unipersonal
1	9	100
2	49	129
3	307	230
4	787	459
5	1494	566
6	2435	1005
7	3616	1182
8	5043	941

Tabla 8.5. *Costo de la MIVC a medida que incrementa el número de individuos en el hogar.*

8.8 COMPARACIÓN DE LAS IMPLEMENTACIONES DE LA MIVC Y DE LA METODOLOGÍA DE FELLEGI Y HOLT.

En censos previos, CANEDIT, una implementación de la metodología de Fellegi y Holt, fue usado para llevar a cabo la validación y la imputación de variables demográficas. Las acciones de imputación de la MIVC y de CANEDIT fueron comparadas en 12,000 registros fallados. Aproximadamente el 98% de éstas tuvieron el mismo número de variables imputadas. La mayoría de las variables restantes tuvo una variable adicional imputada por la MIVC debido a los *edits* más rigurosos de la MIVC basados en la edad más que en la década. CANEDIT usó la década en vez de la edad en los *edits*, porque de lo contrario los costos computacionales son muy grandes.

En algunos casos, la MIVC imputará más del mínimo número de variables si esto produce una acción de imputación más plausible. Esto se ilustra en las siguientes tablas. El registro falla el *edit* que establece que debe haber al menos una diferencia de 15 años entre el padre y el hijo. La imputación CANEDIT incrementa la Edad 1 de 35 a 45 años cambiando la década de nacimiento, haciendo que el *edit* se pase. La MIVC cambia Relación con el jefe

de familia 3 Hija por Esposa y además el Estado civil 3 es cambiado. Esto crea una acción de imputación más plausible que la de CANEDIT.

Registro fallado

Relación con el jefe de familia	*Estado civil*	*Edad*
Jefe	Divorciado	**35**
Hijo	Soltero	8
Hija	Viudo	**36**

Registro imputado por CANEDIT

Relación con el jefe de familia	*Estado civil*	*Edad*
Jefe	Divorciado	**45**
Hijo	Soltero	8
Hija	Viudo	36

Registro imputado por la MIVC

Relación con el jefe de familia	*Estado civil*	*Edad*
Jefe	**Casado**	35
Hijo	Soltero	8
Esposa	**Casado**	36

Las ventajas de la MIVC pueden ser resumidas como sigue:

- Debido a su eficiencia, sus costos crecen, aproximadamente, linealmente al número de *edits*, mientras que, con las aplicaciones de la metodología de Fellegi y Holt, los costos se incrementan exponencialmente.
- Con la MIVC se usan algoritmos relativamente simples, mientras que para implementar la metodología de Fellegi y Holt se requieren sofisticadas técnicas de programación lineal.
- Fellegi y Holt siempre imputa el mínimo número de variables. MIVC imputará ocasionalmente más del mínimo si esto resulta en una acción de imputación más plausible.
- La MIVC puede ser extendida fácilmente a problemas de validación e imputación involucrando únicamente variables numéricas o a problemas que involucran un gran número de variables cuantitativas y cualitativas. La metodología de Fellegi y Holt no es fácilmente extendible por consideraciones computacionales.

8.9 IDENTIFICANDO PAREJAS ANTES DEL PROCESAMIENTO DE LA MIVC.

En 1996, un programa, previo a la MIVC, identificaba parejas potenciales no únicas (e.g., podría haber más de una pareja nuera-hijo en un hogar) al asignar un puntaje a todos los pares de personas en un hogar. Si las relaciones con los jefes de familia, estados civiles,

sexos y edades indican que un par era probablemente una pareja, un puntaje alto era asignado. A las parejas potenciales con los puntajes más altos se les asignaban valores idénticos para una nueva variable: pareja no imputable. Entonces, unas reglas de conflicto, similares a las de la Tabla 8.6 eran aplicadas para determinar si la pareja potencial debía ser retenida. La aplicación de estos *edits* dio como resultado parejas que eran retenidas con características apropiadas (i.e., sexo opuesto, ambos casados, etc.) o parejas que eran descartadas al cambiarse, e.g., la relación con el jefe de familia hijo-nuera por hijo-hija o incluso por inquilino. En algunos casos, la relación con el jefe de familia de una de las personas en parejas potenciales con altos puntajes era puesta en blanco antes de validar e imputar si era inapropiada en términos de la relación con el jefe de familia de la otra persona. Esto daba mayor oportunidad de imputar una relación con el jefe de familia apropiada.

Los *edits* eran especificados en la forma genérica dada en la Tabla 8.6, pero entonces el programa previo a la MIVC expandía las TDBs en un número de réplicas, una por cada posible pareja de personas en el hogar. Con la Tabla 8.1 y un hogar de 6 personas por ejemplo, las variables (1, 2) de la tabla eran reemplazadas por (2, 3), (2, 4), (2, 5), (2, 6), (3, 4), (3, 5), (3, 6), (4, 5), (4, 6), (5, 6). Además, un número igual de permutaciones, i.e., (3, 2), (4, 2), etc., eran generadas para un total de 20 TDBs con $1 \rightarrow 5$ y $2 \rightarrow 6$ indicando, e.g., que los *edits* de la Tabla 8.6 iban a ser aplicados a las personas en la 5ª y 6ª posiciones en el hogar. La necesidad de replicar las TDBs genéricas resulta en el gran número de *edits* listados en la Tabla 8.6 para los hogares más grandes. Las parejas son identificadas antes de validar e imputar para permitir que las parejas más probables sean prioritarias durante la validación e imputación. Hacer esto también reduce el número de *edits*, los cuales, en caso contrario, habrían sido muchos más que el número listado en la Tabla 8.6.

Proposición	***Edit***								
	e_1	e_2	e_3	e_4	e_5	e_6	e_7	e_8	e_9
Relación con el jefe de familia 1 = Hijo/Hija	1	1	1	1	1	1	1	1	0
Relación con el jefe de familia 1 ≠ Hijo/Hija	0	0	0	0	0	0	0	0	1
Relación con el jefe de familia 2 = Hijo/Nuera	1	1	1	1	1	1	1	0	1
Relación con el jefe de familia 2 ≠ Hijo/Nuera	0	0	0	0	0	0	0	1	0
Sexo 1 = Sexo 2	1	0	0	0	0	0	0	0	0
Estado civil 1 = Casado	0	1	0	0	0	0	0	0	0
Estado civil 1 ≠ Casado	0	0	1	0	0	1	0	0	0
Estado civil 2 = Casado	0	0	1	0	0	0	0	0	0
Estado civil 2 ≠ Casado	0	1	0	0	0	0	1	0	0
Estatus de ley común 1 = Sí	0	0	0	1	0	0	0	1	0
Estatus de ley común 1 = No	0	0	0	0	1	1	0	0	0
Estatus de ley común 2 = Sí	0	0	0	0	1	0	0	0	1
Estatus de ley común 2 = No	0	0	0	1	0	0	1	0	0

Tabla 8.6. Edits *entre personas para parejas potenciales "Hijo/Hija-Hijo/Nuera".*

8.10 EXTENSIÓN DE LA MIVC A LA IMPUTACIÓN CUANTITATIVA.

Es útil considerar el extender la MIVC para que lleve a cabo la imputación de un gran número de variables cuantitativas. La MIVC podría entonces procesar muchos más problemas de validación e imputación cuantitativos que los manejados por sistemas tales como GEIS dado el gran costo computacional en determinar el mínimo número de variables a imputar bajo la metodología de Fellegi y Holt. Las restricciones sobre el tipo de *edits* que la MIVC podría manejar serían también mucho menores que las de GEIS.

Un típico conjunto pequeño de *edits* en GEIS expresados en forma de TDBs es dado en las siguientes tablas en forma de reglas de validación y de conflicto.

Proposición	Regla de validación
$V_1 \geq 0$	S
$V_2 \geq 0$	S
$V_3 \geq 0$	S
$V_4 \geq 0$	S

Proposición	Regla de conflicto			
$V_1 \geq 0$	N	-	-	-
$V_2 \geq 0$	-	N	-	-
$V_3 \geq 0$	-	-	N	-
$V_4 \geq 0$	-	-	-	N

La j-ésima proposición numérica $(j = 1,...,4)$ toma la forma:

$$V_j = \sum_{i=1}^{I} b_{ji} V_{ai} - c_j \geq 0$$

donde $V_{ai}, i = 1,...,I$, representa el valor de las I variables numéricas siendo validadas después de que algunas han sido imputadas. b_{ji} y c_j representan constantes.

En esta sección, los *edits* cuantitativos, en la forma de las 4 reglas de conflicto de la tabla anterior, serán discutidos. La MIVC, sin embargo, puede ser extendida para manejar TDBs con proposiciones numéricas con cualquier patrón de 0's y 1's. El mismo enfoque resaltado en las secciones anteriores para la imputación de una mezcla de variables cualitativas y cuantitativas será aplicado aquí. Un vecino más cercano será encontrado y las respuestas en blanco y no válidas junto con las variables esenciales serán imputadas. Si el registro aún falla, los *edits* que ninguna acción de imputación fallaría serán removidos. Entonces el mínimo número posible de acciones de imputación será generado al descartar cualquiera

que no sea una AICCM, cualquiera que no sea esencialmente nueva o cualquiera que siga fallando *edits* específicos sin importar si se imputan variables adicionales.

Con variables cualitativas, un gran número de variables pueden ser frecuentemente descartadas inmediatamente como imputables dado que toman valores idénticos al del vecino más cercano y el registro fallado. Con variables numéricas, es bastante probable que muchas, sino todas, tendrán valores al menos ligeramente diferentes cuando el vecino más cercano y el registro fallado sean comparados. Esto inicialmente hace parecer más difícil de identificar si una proposición numérica es siempre verdadera o siempre falsa dado que el número de posibles acciones de imputación es muy grande. El hecho, sin embargo, de que estamos tratando exclusivamente con variables numéricas en estas proposiciones permite que sean evaluadas muy rápidamente. Esto puede ser visto notando primero que el valor imputado V_{ai} para la i-ésima variable puede ser escrito como:

$$V_{ai} = \ddot{a}_i V_{pi} + (1 - \ddot{a}_i) V_{fi} = \ddot{a}_i (V_{pi} - V_{fi}) + V_{fi}$$

donde:

V_{fi} representa el valor del registro fallado

V_{pi} representa el valor del vecino más cercano

$\ddot{a}_i$ representa una variable indicadora tal que $\ddot{a}_i = 1$ si la i-ésima variable es imputada y $\ddot{a}_i = 0$ si la i-ésima variable no es imputada.

La función V_j en la j-ésima proposición puede entonces ser escrita como:

$$V_j = \sum_i \ddot{a}_i B^*_{ji} - c^*_j$$

donde $B^*_{ji} = b_{ji}(V_{pi} - V_{fi})$ y $c^*_j = c_j - \sum_i b_{ji} V_{fi}$.

Debe notarse que V_j, como una función de $\ddot{a}_i$, permite a las reglas de conflicto ser evaluadas rápidamente a medida que las variables son imputadas de manera secuencial.

El concepto de variables esenciales puede ser generalizado en el caso de reglas de conflicto involucrando solamente variables cuantitativas. Supongamos que la acción de imputación inicial, después de imputar las respuestas en blanco y las no válidas, falla la j-ésima regla de conflicto, es decir, $V^0_j > 0$ donde V^0_j representa el valor de V_j para la acción de imputación inicial. Supongamos además que las acciones de imputación, las cuales pueden ser generadas de la acción de imputación inicial y pasan la j-ésima regla de conflicto, siempre tienen ciertas variables imputadas (sin contar aquellas que fueron imputadas porque estaban en blanco o eran inválidas). Estas variables serán llamadas las ***variables***

esenciales a imputar para la *j*-ésima regla de conflicto. Las variables esenciales a imputar pueden ser identificadas fácilmente calculando primero:

$$\min \left\{ V_j = V_j^0 + \sum_{i-} B_{ji}^* \right\}$$

donde $\sum_{i-} B_{ji}^*$ representa la suma de aquellos valores de B_{ji}^* que son negativos pero sólo para variables que no han sido ya imputadas porque eran invalidas o estaban sin contestar. Se sabe que el $\min \{V_j\} \geq 0$, porque el registro donador pasó el *j*-ésimo *edit*. Entonces determinaremos para cada variable con una B_{ji}^* negativa si el $\min \{V_j - B_{ji}^*\} > 0$. Si esta desigualdad se da, esa variable es una de las variables esenciales porque al eliminarla causará que cualquier otra acción de imputación basada en las otras variables falle la *j*-ésima regla de conflicto.

Supongamos que la acción de imputación inicial aún falla los *edits* después de imputar las respuestas en blanco, las no válidas y las variables esenciales. Deseamos determinar si la *j*-ésima regla de conflicto puede ser descartada dado que el $\max \{V_j\} > 0$ donde $\max \{V_j\}$ representa al máximo valor posible para V_j basado en la acción de imputación inicial más cualquier acción de imputación que pueda ser generada de ella. El que se dé esta relación significa que no existe ninguna acción de imputación que falle el *j*-ésimo *edit* para la pareja registro fallado-donador. Es fácil ver que el $\max \{V_j\} = V_j^0 + \sum_{i+} B_{ji}^*$ donde $\sum_{i+} B_{ji}^*$ representa la suma sobre todos los valores de B_{ji}^* que son positivos pero sólo para aquellas variables que no han sido aún imputadas (porque eran respuestas en blanco, no válidas o variables esenciales). Más tarde en este mismo procesamiento, después de que hayan sido generadas varias acciones de imputación, puede determinarse si el $\min \{V_j\} > 0$ para todas las acciones de imputación que pueden ser generadas de una acción de imputación específica. En esa situación, esta acción de imputación específica puede ser descartada ya que todas las acciones de imputación que puedan ser generadas de ella fallarán la *j*-ésima regla de conflicto.

CONCLUSIONES

1. La metodología de Fellegi y Holt es la base teórica de nuestro trabajo porque resuelve de manera sencilla los problemas que se presentan en la validación e imputación de datos. Además, posee características deseables en todo sistema de validación e imputación tales como: expresar de forma simple y uniforme los edits, imputar a partir de los edits, identificar los campos a imputar (siendo éstos los menos posibles) y mantener las distribuciones marginales de las variables.

2. Al expresar los edits en su forma normal podemos manejar de manera uniforme los diferentes tipos de reglas de validación que actualmente existen. Esto nos permite trabajar con los edits a nivel teórico y dar resultados generales aplicables a cualquier tipo de encuesta.

3. Los Algoritmos GKL1, WW y BCC logran resolver el problema de generar el conjunto completo de edits de manera sistemática y rápida. Dicho problema no fue resuelto por la metodología original de Fellegi y Holt.

4. La elección del algoritmo a utilizar dependerá de las necesidades del usuario y de las características de la encuesta. Por ejemplo, si al aplicar los edits explícitos se tiene que pocos registros fallan edits, podría no ser necesario generar el conjunto completo de edits e imputarlos utilizando el Algoritmo GKL2. De existir muchos registros erróneos lo más indicado sería obtener dicho conjunto para lo cual utilizaríamos el Algoritmo BCC pues éste es una mejora de los Algoritmos GKL1 y WW en cuanto a la rapidez con que logra obtener el conjunto completo de edits. Al

generar este conjunto podemos enfrentarnos a dos situaciones: que se obtenga rápidamente el conjunto o que, después de encontrar los edits implícitos de primer nivel, los implícitos de niveles inferiores se obtengan tan lentamente que nos convenga detenernos en ese momento ya que de continuar con el algoritmo los costos en tiempo se elevarían de tal modo que harían prohibitiva su aplicación completa. En la primera situación, los algoritmos del Capítulo III permiten hacer la imputación de los registros; en la segunda situación, los Algoritmos CW1 y CW2 son los más indicados. Lo anterior permite reducir los costos independientemente de la situación en la que nos encontremos.

5. Aunque al principio la incorporación de los algoritmos al proceso de validación e imputación parezca costosa, los beneficios serán a largo plazo ya que en encuestas posteriores sólo será necesario reespecificar los campos del registro y los edits explícitos para realizar todo este proceso sin más modificaciones.

6. Este trabajo sirve como una base para la generación de programas de computadora para la validación e imputación automática, así también como una fuente de ideas para trabajos posteriores que pretendan estudiar dicho proceso.

7. Nuestro trabajo logra conciliar la experiencia de los especialistas en las encuestas y el potencial que tienen las computadoras. Los primeros intervienen en la especificación de los campos del registro y de los edits explícitos necesarios para la aplicación de los algoritmos propuestos; mientras que, al tener los algoritmos una estructura sencilla, se pueden obtener programas de computadora altamente eficientes.

8. Dado que nuestros ejemplos fueron esencialmente didácticos pues únicamente buscaban verificar que los algoritmos funcionaban, es fundamental la aplicación de éstos a una encuesta real. Además, nuestra teoría no considera la validación e imputación entre registros sino únicamente la que se hace entre los campos de un registro. Por otro lado, no considera errores sistemáticos (no aleatorios) que pueden presentarse en una encuesta en la cual una pregunta haya sido mal diseñada y por lo tanto contestada incorrectamente por una gran parte de los encuestados. Esto último deberá minimizarse mediante la revisión exhaustiva del cuestionario de la encuesta.

9. Es importante resaltar que la teoría no funciona bien con las variables continuas ya que ésta no fue hecha ex profeso para incluirlas. Trabajos posteriores deberán considerarlas con mayor amplitud. Análogamente ocurre esto con los blancos por pase.

BIBLIOGRAFÍA

Banker, M. (1991); *Alternative Method of Doing Quantitative Variable Imputation*; Statistics Canada Memorandum.

Bankier, M.; Houle, A. M.; Luc, M. y Newcombe, P. (1997); "1996 Canadian Census Demographic Variables Imputation"; *Proceedings of the Section on Survey Research Methods Section*; American Statistical Asociation, pp. 389-394.

Bankier, M. (1999); *Experience with the New Imputation Methodology Used in the 1996 Canadian Census with Extensions for Future Censuses*; Statistical Commission and Working Paper No. 24; Economic Commission for Europe; Conference of European Statisticians, UN/ECE Work Session on Statistical Data Editing; http://www.unece.org/stats/documents/1999/06/sde/24.e.pdf

Bankier, M.; Lachance, M. y Poirier, P. (2000); *2001 Canadian Census Minimum Change Donor Imputation Methodology*; Statistical Commission and Working Paper No. 17; Economic Commission for Europe; Conference of European statisticians, UN/ECE Work Session on Statistical Data Editing; http://www.unece.org/stats/documents/2000/10/sde/17.e.pdf

Chen, B.-C. (1998); *Set Covering Algorithms in Edit Generation*; Bureau of the Census Statistical Research Division; Statistical Research Report Series No. RR98/06; U.S. Bureau of the Census; Statistical Research Division; Washington D.C.; http://www.census.gov/srd/papers/pdf/rr98-06.pdf

Fellegi, I. P. y Holt, D. (1976); "A Systematic Approach to Automatic Edit and Imputation"; *Journal of the American Statistical Association* **71**: 17-35.

Garfinkel, R. S. y Nemhauser, G. L. (1972); *Integer programming*; New York: J. Wiley; 427 p.

Garfinkel, R. S.; Kunnathur, A. S. y Liepins, G. E. (1986); "Optimal Imputation of Erroneous Data: Categorical Data, General Edits"; *Operations Research* **34**: 744-751.

Subcommittee on Data Editing in Federal Statistical Agencies (1990); *Data Editing in Federal Statistical Agencies*; Federal Committee on Statistical Methodology; Statistical Policy Working Paper 18; Washington D.C.: Office of Management & Budget.

Winkler, W. E. (1995); *Editing Discrete Data*; Bureau of the Census, Washington D.C.; http://www.amstat.org/sections/srms/proceedings/papers/1995_016.pdf

Winkler, W. E. (1997); *Set-Covering and Editing Discrete Data*; Bureau of the Census, Washington D.C.; http://www.census.gov/srd/www/abstract/rr9801.html

Winkler, W. E. y Chen, B.-C. (2002); *Extending the Fellegi-Holt Model of Statistical Data Editing*; Statistical Research Division; U.S. Bureau of the Census; Washington D.C.; http://www.census.gov/srd/papers/pdf/rrs2002-02.pdf

Bibliografía adicional

Brant, J. D. y Chalk, S. M. (1985); "The Use of Automatic Editing in the 1981 Census"; *Journal of the Royal Statistical Society* **148**: 126-146.

Freund, R. J. y Hartley, H. O. (1967); "A Procedure for Automatic Data Editing"; *Journal of the American Statistical Association* **62**: 341-352.

Giles, P. (1988); "A Model for Generalized Edit and Imputation of Survey Data"; *The Canadian Journal of Statistics* **16** Supplement: 57-73.

Nordbotten, S. (1995); "Editing Statistical Records by Neural Networks"; *Journal of Official Statistics* **11**: 391-411.

Oshungade, I. O. (1989); "Some Methods of Handling Item Non-response in the Categorical Data"; *The Statistician* **38**: 281-296.

Pierzchala, M. (1990); "A Review of the State of the Art in Automated Data Editing and Imputation"; *Journal of Official Statistics* **56**: 355-377.

Pritzker, L.; Ogus, J. y Hansen, M. H. (1965); "Computer Editing Methods: Some Applications and Results"; *Bulletin of the International Statistical Institute, Proceedings of 35th Session* **41**: 417-441.

Pullum, T. W; Harpham, T. y Ozsever, N. (1986); "The Machine Editing of Large-Sample Surveys: The Experience of the World Fertility Survey"; *International Statistical Review* **54**: 311-326.

Szameitat, K. y Zindler, H.-J. (1965); "The Reduction of errors in Statistics by Automatic Corrections"; *Bulletin of the International Statistical Institute, Proceedings of 35th Session* **41**: 395-417.

Yates, F. (1971); "The use of Computers for Statistical Analysis: A Review of Aims and Achievements"; *Bulletin of the International Statistical Institute, Proceedings of 38th Session* **44**: 39-53.