# UN ALGORITMO APROXIMADO PARA EL PROBLEMA DE COBERTURA MÍNIMA DE COSTO ÚNICO

## Approximation algorithm for unicost cover problem

**M.Sc. Lucio Torrico**
luciotorrico@gmail.com

## RESUMEN

En general las soluciones exactas para el unicost set cover problem se obtienen a través de algoritmos intratables pues el problema es NP. Hay diferentes algoritmos aproximados de tipo polinomial. Por ejemplo el conocido algoritmo estándar de Johnson que utiliza la técnica Greedy. Presentamos un algoritmo aproximado que implementa la reducción de redundancias, la utilización de disyunciones y una métrica, para deseleccionar subconjuntos, en base a sus deméritos. El mismo parece comportarse bastante bien para los experimentos realizados, en comparación con el estándar e incluso con otros algoritmos aproximados.

**Palabras clave:**

algoritmo aproximado; cobertura mínima; cobertura mínima de costo único

## *ABSTRACT*

Exact solutions for unicost set cover problem generally are obtained through intractable algorithms, since the problem is NP. There are different approximation algorithms of polynomial type. By example the Johnson's standard algorithm that uses Greedy technique.

Here we present an approximation algorithm that implements redundancy reduction, using disjunctions and a metric, for deselect subsets, based on their demerits.

This algorithm seems to perform well in the experiments we perform, compared to the standard algorithm and even other approximate algorithms.

***keywords:***

approximation algorithm, set cover, unicost set cover

## INTRODUCCIÓN

El problema de cobertura mínima (y un problema asociado conocido como el problema de cobertura exacta) son conocidos en la algorítmica por ser NP (Cormen, Leiserson, Rivest & Stein, 2002).

Un algoritmo aproximado halla una cuasi-solución (cuando no se puede hallar la solución exacta por ser prohibitiva en tiempo).

Presentamos el problema, un algoritmo aproximado estándar, una nueva propuesta y el desempeño de esta última.

## EL PROBLEMA

Dados los conjuntos de objetos
$S_1, S_2, \ldots, S_n$.
Por ejemplo:
$S_1=\{O_1\}$
$S_2=\{O_2, O_3\}$
$S_3=\{O_3, O_4\}$
$S_4=\{O_2, O_4\}$
$S_5=\{O_4\}$
Podemos formar un universo U, uniendo todos esos conjuntos.
$U = S_1 \cup S_2 \cup \ldots \cup S_n$
En el ejemplo: $U=\{O_1, O_2, O_3, O_4\}$

O bien podemos partir de un universo U y de un conjunto $\{S_1,S_2,\ldots,S_n\}$ de subconjuntos de U (cuya unión es el universo).

Es claro que podemos trabajar sólo con los subíndices.

En el ejemplo: U={1,2,3,4}

$S_1$={1}
$S_2$={2,3}
$S_3$={3,4}
$S_4$={2,4}
$S_5$={4}

El problema de cobertura mínima consiste en seleccionar el menor número de subconjuntos tales que su unión devuelva el universo.
En el ejemplo $S_1,S_2$ no es solución pues $S_1 \cup S_2 \neq U$.
En el ejemplo $S_1,S_2,S_3,S_4$ no es solución pues aunque su unión es U, no es mínimo (tiene cuatro subconjuntos). Hay otra selección con menos subconjuntos (por ejemplo $S_1,S_2,S_4$)

Hay otro problema de cobertura mínima con pesos, donde cada $S_i$ tiene un peso o costo asociado, y donde se trata de elegir la cobertura con el menor costo posible.
No consideraremos dicho problema. (Es posible ver la cobertura mínima que estamos considerando como si todos los subconjuntos tuvieran costo/peso igual a 1 –unicost set cover-. Nosotros ignoraremos el asunto de los pesos/costos).

Un enunciado formal es:
Dado el universo U y un conjunto $\{S_1,S_2,\ldots,S_n\}$ de subconjuntos de U (cuya unión es el universo), hallar el menor número de subconjuntos $S_{i1},S_{i2},\ldots,S_{ik}$ tal que $\cup S_{ij} = U \quad (1 \leq j \leq k)$

## EL ALGORITMO ESTÁNDAR DE JOHNSON

Johnson proporciona una cuasi-solución bastante aproximada al mínimo a través de la técnica greedy (Shekhar, 2009).

Entrada:Un conjunto universo U
Un grupo de subconjuntos $S_i$
Salida: Una cobertura mínima para U
sol = {}
No_Cubierto = U
MIENTRAS No_Cubierto ≠ {}
Seleccionar el más grande $S_j$
sol = sol U $S_j$
No_Cubierto = No_Cubierto - $S_j$
$S_i = S_i - S_j$
FinMIENTRAS

Ejemplo:
U = {1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12}
$S_1$ = {1, 2, 3, 4, 5, 6}
$S_2$ = {5, 6, 8, 9}
$S_3$ = {1, 4, 7, 10}
$S_4$ = {2, 5, 7, 8, 11}
$S_5$ = {3, 6, 9, 12}
$S_6$ = {10, 11}

Seleccionamos $S_1$
sol = $S_1$
No_Cubierto = {7, 8, 9, 10, 11, 12}
$S_2$ = {8, 9}
$S_3$ = {7, 10}
$S_4$ = {7, 8, 11}
$S_5$ = {9, 12}
$S_6$ = {10, 11}

Seleccionamos $S_4$
sol = $S_1$, $S_4$
No_Cubierto = {9, 10, 12}
$S_2$ = {9}
$S_3$ = {10}
$S_5$ = {9, 12}
$S_6$ = {10}

Seleccionamos $S_5$
sol = $S_1$, $S_4$, $S_5$
No_Cubierto = {10}
$S_2$ = { }
$S_3$ = {10}
$S_6$ = {10}
Seleccionamos $S_6$. **sol** = $S_1$, $S_4$, $S_5$, $S_6$
No_Cubierto = { }
Y terminamos, con dicha solución aproximada.

Nótese que $S_3$, $S_4$, $S_5$ es mejor.

Hay unas optimizaciones propuestas por Duh and Furer y sobre ellas otras más finas de Asaf Levin, mismas que ofrecen una solución aproximada con a lo más (en el caso peor) el doble de subconjuntos que la solución mínima (Shekhar, 2009).

Hay otras maneras de enfocar el hallazgo de buenas aproximaciones al mínimo a través de la programación lineal o los algoritmos genéticos (Shekhar, 2009) (Young, 2008) (Beasley, 1996).

Grossman and Wohl -mencionados en (Musliu, 2006)- sugieren que el mejor algoritmo es el randomized greedy: Como el de Johnson, pero si hay varios subconjuntos más grandes, seleccionar uno al azar.

## EL ALGORITMO PROPUESTO

Presentaremos algunas ideas previas para mejor comprensión.

Sea $\cup S_i$ la unión de todos los subconjuntos que estamos considerando actualmente.

Llamaremos **subconjunto imprescindible** al subconjunto $S_j$, si $[\cup S_i - S_j] \neq U$

En el ejemplo anterior $S_5$ es imprescindible, pues no hay otro

subconjunto que cubra el elemento 12.

Si describimos el problema a través una matriz binaria, donde los elementos que faltan por cubrir del universo son las columnas, y los subconjuntos candidatos son las filas, el ejemplo anterior se ve así:

| | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| $S_1$ | 1 | 1 | 1 | 1 | 1 | 1 | | | | | | |
| $S_2$ | | | | | 1 | 1 | | 1 | 1 | | | |
| $S_3$ | 1 | | | 1 | | | 1 | | | 1 | | |
| $S_4$ | | 1 | | | 1 | | 1 | 1 | | | 1 | |
| $S_5$ | | | 1 | | | 1 | | | 1 | | | 1 |
| $S_6$ | | | | | | | | | | 1 | 1 | |

En las columnas con un único "1" vemos en qué fila se halla dicho "1", el subconjunto correspondiente **es** imprescindible.

Es obvio que los subconjuntos imprescindibles, deben estar en la solución.

Cada vez que se incluya un subconjunto en la solución, podemos eliminar los elementos del mismo, tanto del universo cómo de los demás subconjuntos, pues ya están cubiertos. Llamaremos a esto **redimensionar el problema**.

En el ejemplo anterior, al incluir $S_5$ en la solución, la redimensión daría:

| | 1 | 2 | 4 | 5 | 7 | 8 | 10 | 11 |
|---|---|---|---|---|---|---|---|---|
| $S_1$ | 1 | 1 | 1 | 1 | | | | |
| $S_2$ | | | | 1 | | 1 | | |
| $S_3$ | 1 | | 1 | | 1 | | 1 | |
| $S_4$ | | 1 | | 1 | 1 | 1 | | 1 |
| $S_6$ | | | | | | | 1 | 1 |

Llamaremos **elemento o columna redundante** a una columna tal que sus elementos incluyen a todos los elementos de otra columna.

En la última matriz de ejemplo la columna **2** está incluida en la columna **5** (la columna **5** incluye a la columna **2**), es decir, llamaremos redundante a la columna **5**.

Como otro ejemplo, dado que **1** está incluido en **4**, la columna **4** es redundante.

| | ... | Y | ... | X | ... |
|---|---|---|---|---|---|
| $S_{..}$ | | 1 | | | |
| . | . | 1 | . | 1 | . |
| . | . | 1 | . | 1 | . |
| . | . | | . | | . |
| | | 1 | | 1 | |
| | | 1 | | | |

Dado que la columna Y incluye a la columna X, Y es redundante.

En efecto: Debemos cubrir el elemento **X**. Eso sólo puede hacerse con uno de los subconjuntos que corresponden a aquellos donde hay un "1" en la mencionada columna **X**.

Como dicha columna, está incluida en la columna **Y**, al cubrir el elemento **X** (digamos con el subconjunto $S_j$) también cubriremos -automáticamente- el elemento **Y**.

Debido a ello, las columnas redundantes pueden eliminarse.

Al eliminar las columnas redundantes **4** y **5**, el ejemplo quedaría así:

| | 1 | 2 | 7 | 8 | 10 | 11 |
|---|---|---|---|---|---|---|
| $S_1$ | 1 | 1 | | | | |
| $S_2$ | | | | 1 | | |
| $S_3$ | 1 | | 1 | | 1 | |
| $S_4$ | | 1 | 1 | 1 | | 1 |
| $S_6$ | | | | | 1 | 1 |

Llamaremos **subconjunto redundante** a la fila que está incluida en otra.

Es claro, que si $S_i \subseteq S_j$, todos los elementos que podría cubrir $S_i$ estarán cubiertos por $S_j$.

Cualquier solución que requiera de $S_i$, puede funcionar igual tomando en su lugar $S_j$.

Las filas redundantes pueden eliminarse. Más adelante, ampliaremos un poco esta idea.

En el ejemplo $S_2 \subseteq S_4$. Eliminando $S_2$ todo queda así:

| | 1 | 2 | 7 | 8 | 10 | 11 |
|---|---|---|---|---|---|---|
| $S_1$ | 1 | 1 | | | | |
| $S_3$ | 1 | | 1 | | 1 | |
| $S_4$ | | 1 | 1 | 1 | | 1 |
| $S_6$ | | | | | 1 | 1 |

Es evidente que se trata de reducir la matriz original ya sea por filas, ya sea por columnas.

Aún cuando sea una fila/columna a la vez.

Hay otro par de ideas cuyo éxito está en experimentación, aunque parecen ser prometedoras en algún grado. Deben manejarse según se indica.

Un mecanismo para disminuir filas explota la idea de **disyunción**.

Si $S_i \cap S_j = \{\}$, entonces eliminaremos ambos subconjuntos y añadiremos uno nuevo resultante de su unión.

| | 1 | 2 | 3 | 4 |
|---|---|---|---|---|
| $S_1$ | 1 | 1 | | |
| $S_2$ | | | 1 | 1 |
| $S_3$ | 1 | | 1 | |
| $S_4$ | | 1 | | 1 |

Ejemplo:
$S_1 \cap S_2 = \{\}$, los eliminamos y añadimos su unión bajo el nombre de $S_5$:

| | 1 | 2 | 3 | 4 |
|---|---|---|---|---|
| $S_3$ | 1 | | 1 | |
| $S_4$ | | 1 | | 1 |
| $S_5$ | 1 | 1 | 1 | 1 |

Nótese que hemos resaltado (con una línea punteada) el hecho de que $S_5$ es un subconjunto que forma parte de otro grupo (no de los originales).

Llamaremos **supracardinalidad (sc)** de un subconjunto al número de subconjuntos que lo componen.

Los subconjuntos originales tienen supracardinalidad igual a 1.

En el ejemplo $S_5$ tienen supracardinalidad igual a 2.

*La detección de inclusión de una fila en otra se hará sólo dentro de un mismo grupo (donde todos los subconjuntos tienen la misma* **sc***).*

Este proceso puede añadir varios subconjuntos nuevos y es posible que con estas adiciones, suceda que $S_i = S_j$ (*incluso si ambos subconjuntos están en diferentes grupos*).

Si sucede, eliminaremos el subconjunto con mayor supracardinalidad.

De modo mnemotécnico el algoritmo inicial hasta ahora es:

Imprescindibilidad
Inclusión columnas
Igualdad subconjuntos
Inclusión filas
Disyunciones

Resta decidir qué hacer cuando no hayan estos aspectos:

Es decir, no hay imprescindibles, no se puede eliminar filas o columnas redundantes, no existen conjuntos iguales, ni disjuntos.

Debemos elegir cuál fila dejar de tomar en cuenta (o bien cuál incluir en la solución).

En tal caso, haremos la selección así:

**Guardamos la matriz actual**.

**Elaboramos un ranking** de prescindibilidad/ imprescindibilidad así:

De una columna con un mínimo de "1"'s:

**Con cada subconjunto** que tenga un "1" en dicha columna y empezando en la matriz actual (Nótese que multiplicamos o dividimos los incrementos o decrementos por la supracardinalidad del subconjunto considerado):

- Supondremos su eliminación (por este motivo asignaremos +1/sc a este subconjunto) (y redimensionaremos el problema).
- Aplicamos recurrentemente el algoritmo inicial hasta que no puedan eliminarse más subconjuntos:

En este proceso:
A los subconjuntos que resultaren imprescindibles les asignaremos +2/sc.
A los subconjuntos que resultaren eliminados por ser redundantes les asignaremos -1×sc.
A los subconjuntos que quedaren (si es que los hay) les asignaremos +1/sc.

Con todos los incrementos y decrementos del anterior proceso (para todos los subconjuntos en consideración) elaboramos un acumulado:

Se elimina de la matriz guardada un subconjunto que tenga el mínimo acumulado.

Y volvemos a comenzar con el algoritmo inicial.

Se presenta un ejemplo luego de exponer el algoritmo completo:

Entrada: Un conjunto universo U
Un grupo de subconjuntos $S_i$
Salida: Una cobertura mínima para U
REPETIR
Case:
Existen subconjuntos imprescindibles:
Añadirlos a la solución (redimensión)
Existe inclusión columnas col**X** ⊆ col**Y**:
Eliminar col**Y** (no hay redimensión)
Existe igualdad de filas $S_i = S_j$:
Eliminar el subconjunto de mayor supracardinalidad
Existe inclusión filas $S_i \subseteq S_j$ (en un mismo grupo): Eliminar $S_i$ (no hay redimensión)
Existe una disyunción: Eliminar los subconjuntos involucrados añadir el nuevo subconjunto (registrar su supracardinalidad y añadirlo en el grupo adecuado) (no hay redimensión)
En otro caso:
En una columna con menos "1"'s:
Elaborar el ranking y el acumulado
Y eliminar un subconjunto que tenga el acumulado mínimo (sin redimensión)
FinCase
HASTA U = {}
Deshacer las uniones de subconjuntos disjuntos de la solución
Realizar un último cálculo de imprescindibilidad con esos subconjuntos
Para la Solución final incluir sólo los subconjuntos imprescindibles.
(Una variante es guardar como solución alterna las uniones de disjuntos que obtengan el universo.
Y al final, luego de deshacer las uniones de subconjuntos disjuntos, realizar un último cálculo de imprescindibilidad y quedarnos sólo con los subconjuntos imprescindibles, comparar las soluciones alternas con la obtenida y quedarse con la menor)

**Ejemplo**

| | 1 | 2 | 3 | 4 | 5 | 6 | 7 |
|---|---|---|---|---|---|---|---|
| $S_1$ | 1 | | 1 | 1 | | 1 | 1 |
| $S_2$ | | 1 | 1 | | 1 | | 1 |
| $S_3$ | 1 | 1 | | | | | |
| $S_4$ | | | 1 | 1 | 1 | | |
| $S_5$ | | | | 1 | 1 | | |
| $S_6$ | | | | | 1 | 1 | |
| $S_7$ | | 1 | | | | 1 | 1 |

No hay imprescindibles
No hay inclusión de columnas
$S_5 \subseteq S_4$
Eliminamos $S_5$

| | 1 | 2 | 3 | 4 | 5 | 6 | 7 |
|---|---|---|---|---|---|---|---|
| $S_1$ | 1 | | 1 | 1 | | 1 | 1 |
| $S_3$ | | 1 | 1 | | 1 | | 1 |
| $S_4$ | 1 | 1 | | | | | |
| $S_5$ | | | | 1 | 1 | 1 | |
| $S_6$ | | | | | | 1 | 1 |
| $S_7$ | | 1 | | | | 1 | 1 |

No hay imprescindibles

col4 $\subseteq$ col3

Eliminar col3

| | 1 | 2 | 4 | 5 | 6 | 7 |
|---|---|---|---|---|---|---|
| $S_1$ | 1 | | 1 | | 1 | 1 |
| $S_3$ | | 1 | | 1 | | 1 |
| $S_4$ | 1 | 1 | | | | |
| $S_5$ | | | 1 | 1 | | |
| $S_6$ | | | | 1 | 1 | |
| $S_7$ | | 1 | | | 1 | 1 |

No hay imprescindibles

No hay inclusión de columnas

No hay inclusión de filas

$S_1 \cap S_4 = \{\}$

Eliminar $S_1$, $S_4$. Añadir $S_8$ con **sc**=2

| | 1 | 2 | 4 | 5 | 6 | 7 |
|---|---|---|---|---|---|---|
| $S_1$ | 1 | | 1 | | 1 | 1 |
| $S_3$ | | 1 | | 1 | | 1 |
| $S_5$ | | | | 1 | 1 | |
| $S_7$ | | 1 | | | 1 | 1 |
| $S_8$ | 1 | 1 | 1 | 1 | | |

No hay imprescindibles

col1 $\subseteq$ col4

Eliminar col4

| | 1 | 2 | 5 | 6 | 7 |
|---|---|---|---|---|---|
| $S_1$ | 1 | | | 1 | 1 |
| $S_3$ | | 1 | 1 | | 1 |
| $S_5$ | | | 1 | 1 | |
| $S_7$ | | 1 | | 1 | 1 |
| $S_8$ | 1 | 1 | 1 | | |

No hay imprescindibles

No hay inclusión de columnas

No hay inclusión de filas

No hay disyunciones

Elegimos la columna **1** como aquella con menos "1"'s

Para cada subconjunto con un "1" en dicha columna ($S_1$,$S_8$) y empezando en la matriz actual, suponemos su eliminación y aplicamos recurrentemente el algoritmo preliminar:

Por claridad, **a la derecha colocaremos los incrementos o decrementos**.

Para $S_1$: **+1/1=1 a $S_1$**

| | 2 | 5 |
|---|---|---|
| $S_3$ | 1 | 1 |
| $S_4$ | | 1 |
| $S_5$ | 1 | |
| $S_8$ | 1 | 1 |

No hay imprescindibles; ni inclusión de columnas; ni igualdad ni inclusión de filas.

$S_6 \subseteq S_5$ y $S_7 \subseteq S_5$: Eliminamos $S_6$,$S_7$ **-1×1=-1 a $S_6$, -1×1=-1 a $S_7$**

| | 2 | 5 |
|---|---|---|
| $S_3$ | 1 | 1 |
| $S_8$ | 1 | 1 |

No hay imprescindibles. col**2** $\subseteq$ col**5**: Eliminar col**5**

| | 2 |
|---|---|
| $S_3$ | 1 |
| $S_8$ | 1 |

$S_2 = S_8$ : Eliminamos $S_8$ **-1×2=-2 a $S_8$**

| | 2 |
|---|---|
| $S_3$ | 1 |

$S_3$ es imprescindible. Lo eliminamos **+2/1=2 a $S_2$**

No hay más subconjuntos que puedan eliminarse.

Para $S_8$: **+1/2=0.5 a $S_8$**

| | 6 | 7 |
|---|---|---|
| $S_1$ | 1 | 1 |
| $S_3$ | | 1 |
| $S_5$ | 1 | |
| $S_7$ | 1 | 1 |

No hay imprescindibles; ni inclusión de columnas.

$S_1 = S_7$ : Eliminamos $S_1$ **-1×1=-1 a $S_1$**

| | 6 | 7 |
|---|---|---|
| $S_3$ | | 1 |
| $S_5$ | 1 | |
| $S_7$ | 1 | 1 |

No hay imprescindibles; ni inclusión de columnas; ni igualdad de filas.

$S_3 \subseteq S_7$ y $S_5 \subseteq S_7$: Eliminamos $S_3$,$S_5$ **-1×1=-1 a $S_3$, -1×1=-1 a $S_5$**

| | 2 | 5 |
|---|---|---|
| $S_7$ | 1 | 1 |

$S_7$ es imprescindible. Lo eliminamos **+2/1=2 a $S_7$**

No hay más subconjuntos que puedan eliminarse.

El acumulado es:

| $S_1$ | $S_2$ | $S_5$ | $S_7$ | $S_8$ |
|---|---|---|---|---|
| 0 | 1 | -2 | 1 | -1.5 |

De la matriz original eliminamos $S_6$, sin redimensionar:

| | 1 | 2 | 5 | 6 | 7 |
|---|---|---|---|---|---|
| $S_1$ | 1 | | | 1 | 1 |
| $S_3$ | | 1 | 1 | | 1 |
| $S_7$ | | 1 | | 1 | 1 |
| $S_8$ | 1 | 1 | 1 | | |

No hay imprescindibles

col**5** $\subseteq$ col**2**

Eliminar col**2**

| | 1 | 5 | 6 | 7 |
|---|---|---|---|---|
| $S_1$ | 1 | | 1 | 1 |
| $S_3$ | | 1 | | 1 |
| $S_7$ | | | 1 | 1 |
| $S_8$ | 1 | 1 | | |

No hay imprescindibles

col**6** $\subseteq$ col**7**

Eliminar col**7**

| | 1 | 5 | 6 |
|---|---|---|---|
| $S_1$ | 1 | | 1 |
| $S_3$ | | 1 | |
| $S_7$ | | | 1 |
| $S_8$ | 1 | 1 | |

No hay imprescindibles

No hay inclusión de columnas

$S_7 \subseteq S_1$

Eliminamos $S_7$

| | 1 | 5 | 6 |
|---|---|---|---|
| $S_1$ | 1 | | 1 |
| $S_3$ | | 1 | |
| $S_8$ | 1 | 1 | |

$S_1$ es imprescindible

lo añadimos a la

SOL= $S_1$

Redimensionamos

| | 5 |
|---|---|
| $S_3$ | 1 |
| $S_8$ | 1 |

$S_2 = S_8$ : Eliminamos $S_8$

| | 5 |
|---|---|
| $S_3$ | 1 |

$S_3$ es imprescindible lo añadimos a la

SOL= $S_1$,$S_3$ Redimensionamos

$U = \{\}$

No hay uniones que deshacer.

Una verificación en la matriz original muestra que ambos son imprescindibles.

(Y no hay soluciones alternas con las que comparar). Luego, la SOL= $S_1$,$S_2$.

## DESEMPEÑO

Las pruebas se hicieron en un script de Matlab (m-file).

Se han elaborado varios ejemplos de tamaño modesto (un máximo de 12 subconjuntos y un máximo de 15 elementos) que podemos asemejar a lo que en otros contextos se conoce como ejemplos canónicos.

El algoritmo ha solucionado con satisfacción dichos ejemplos y hallado soluciones cualitativamente mejores que el de Johnson, a costa de más de tiempo de cálculo debido a su naturaleza heurística y aproximada.

También se ha contrastado el algoritmo con datos de prueba generalmente aceptados –existentes en la nube, en particular la OR-Library (Beasley, 1990)- y se ha comportado bien.

Por simplificación consideraremos el número de filas n igual al de columnas m.

Es claro que la detección de inclusiones y disyunciones toma un tiempo proporcional a $n^2$ (pues cada fila/columna se compara con las otras), al que hay que añadir la comparación uno a uno de cada elemento lo que añade un tiempo de n, haciendo un total de $n^3$. Ello se hace n veces, eliminado una fila/columna a la vez, en el peor de los casos. Dando un total de $n^4$ operaciones.

En el subcaso de detección del subconjunto menos meritorio, se hace exactamente lo mismo con la matriz resultante de eliminar un subconjunto, para cada subconjunto de la columna con menos unos: Es decir, n veces $n^4$ operaciones: Haciendo un total de $n^5$ operaciones.

Total $O(n^5)$. Complejidad polinomial.

Hay quienes critican el costo en tiempo, debido a los muchos cálculos de este tipo de soluciones.

Sin embargo, la persecución de respuestas cualitativamente mejores que el estándar, e incluso que sus mejoras, hace que valga la pena pagar este costo (polinomial -no exponencial-).

Lamentablemente cuando el número de datos tiende a subir (que era el objetivo de esta aproximación) a la demora se agrega que la calidad tiende a desaparecer e incluso a hacerse negativa.

## CONCLUSIONES

Se ha presentado el problema de cobertura mínima de costo único, el algoritmo estándar y una alternativa de algoritmo.

La eliminación de redundancias, junto al uso de la disyunción y el uso de la métrica para elegir –con una argumentación plausible- cuál subconjunto dejar de tomar en cuenta para la solución, presentan un comportamiento interesante, incluso a pesar del (no excesivamente) mayor tiempo que toma respecto de otras aproximaciones.

El heurístico parece bueno y puede funcionar mejor, quizás combinándolo con el conocido análisis de vecindad o afinarlo con otros decrementos/incrementos.

Es, en todo caso, una aproximación diferente de las otras que buscan seleccionar cuál subconjunto añadir a la solución: En este caso se busca cuál no incluir en ella por sus deméritos.

## BIBLIOGRAFÍA

Beasley J.E., Chu P.C. (1996). "*A Genetic Algorithm for the Set Covering Problem*". In *European Journal of Operational Research , 1996*

Cormen Thomas, Leiserson Charles, Rivest Ronald, Stein Clifford. (2002). "*Introduction to Algorithms*". Segunda Edición. McGraw-Hill.

Musliu Nysret. (2006). "*Local search algorithm for unicost set covering problem* ". IEA/AIE, volume 4031of *Lecture Notes in Computer Science, page 302-311.*

Shekhar Himanshu. (2009). "*Survey of approximation algorithms for set cover problem*". University of North Texas. Thesis.

Young Neal. (2008). "*Greedy Set-Cover Algorithms*". In *Encyclopedia of Algorithms, Springer.*

Beasley J.E. (1990). Actualizado en 2012. http://people.brunel.ac.uk/~mastjjb/jeb/info.html. (Consultado Agosto de 2013)