# APLICACION DE LA BUSQUEDA TABU EN LA CLASIFICACION POR PARTICIONES

Alex Murillo Fernández, CIMPA, Universidad de Costa Rica, Sede del Atlántico

**RESUMEN**
Se presenta un método de análisis de conglomerados (clasificación automática) utilizando la técnica de optimización combinatoria llamada búsqueda tabú, para obtener clases bien separadas entre ellas y lo más homogéneas posibles. El algoritmo se propone encontrar la partición óptima de un conjunto de individuos dependiendo del criterio de inercia o varianza intra-clases, tratando de escapar de la optimalidad local. Se presentan dos versiones del método: el original, que introduce el valor de la inercia en la lista tabú, y el mejorado, que penaliza solo algunas características la partición. Se marcan las diferencias y se hace un análisis comparativo entre estos.

**Palabras clave**: análisis de conglomerados, clasificación automática, optimización combinatoria, clasificación optimal, inercia intra-clase, movimiento prohibido, búsqueda tabú.

MSc: 68R05

**ABSTRACT**
We present an improved method for clustering by using the combinatorial optimization technique called tabu search, for obtaining homogeneous and well-separated classes. The algorithm intends to find the optimal partition of a set of objects from the point of view of the within-classes variance criterion, trying to escape from local minima. Two versions of the method are presented: the original one, that introduces the variance value in tabu list, and the improved one, that penalizes only some partition features. Differences and comparisons are pointed out.

**Key words**: cluster analysis, automatic classification, combinatorial optimization, optimal classification, within classes variance, forbidden move, tabu search.

## 1. INTRODUCCION

Los problemas que trata la optimización combinatoria pueden definirse como sigue: *dado un conjunto finito y una función de costo sobre cada elemento de ese conjunto, se busca optimizar la función de costos sobre los elementos de ese conjunto,* y la optimización depende del planteamiento del problema, esto es, si se desea maximizar o minimizar la función de costos. Todos estos problemas de optimización combinatoria tienen un método que se deduce de su fácil presentación; este método es: evaluar la función de costos en cada elemento del conjunto finito y escoger el elemento que optimiza la función. Sin embargo, cuando el conjunto es sensiblemente grande, el método mencionado se ve limitado en la práctica, por dificultades de cómputo y es por esta razón que la optimización combinatoria haya tenido un gran desarrollo teórico en los últimos años, principalmente en el campo de la investigación de operaciones.

En Análisis de Datos surgen a menudo problemas de optimización combinatoria, como es el caso de la Clasificación Automática, la regresión no lineal, el análisis de proximidades (posicionamiento multidimensional), las rotaciones varimax oblicuas, entre otros ejemplos.

Desde el punto de vista de la clasificación automática por particionamiento, el conjunto finito que mencionamos consiste en todas las particiones de un conjunto de individuos $\Omega$ y la función de costo es una función que mida la adecuación de la partición. La función de costo más usada en las aplicaciones es la inercia intra-clase, aunque no es la única.

Para tener una idea de lo que esto significa desde el punto de vista combinatorio, si se denota por $n$ el número de individuos de $\Omega$ y $k$ el número de clases de la partición que se busca, entonces el número de particiones de $\Omega$ en $k$ clases es:

$$S(n,k) = \frac{1}{k!}\sum_{i=0}^{k}(-i)^{k-i}\,\frac{k!}{i!\,(k-i)!}\,i^n .$$

Así, el número de particiones no vacías de $\Omega$ (es decir, el cardinal del conjunto finito que mencionamos antes) sería:

$$B(n) = \exp(-1)\sum_{i=0}^{\infty}\frac{(i+1)^n}{i!}.$$

Poniendo algunos números conservadores en las fórmulas anteriores:

$S(60,2) \approx 10^{18}$, $S(100,5) \approx 10^{68}$;

$B(10) = 115945$, $B(15) = 1382958545$, $B(40) \approx 1.57 \times 10^{35}$.

La optimización combinatoria determinística ya ha sido tratada de usar en problemas de particionamiento, por varios autores [Cailliez-Pagés (1976), Glover-Taillard (1993) y Murillo (1996)]. Nosotros nos proponemos la implementación de un método estocástico, que ha sido planteado en los últimos años. Este método tiene la ventaja de que teóricamente mejor para encontrar los óptimos globales de los problemas, de ahí nuestro interés por estudiar este método. En este artículo se enfoca el problema de clasificación automática por particiones por el método estocástico de búsqueda tabú.

## 2. BUSQUEDA TABU

Fred Glover [1993] propuso, a principios de los años 70, un procedimiento heurístico llamado búsqueda tabú, el cual se utiliza con gran éxito para resolver problemas de optimización combinatoria, cuya característica principal es la de "escapar" de la optimalidad local.

Los métodos clásicos utilizados en problemas combinatorios consisten en dos fases: construcción y mejoramiento; la búsqueda tabú se incorpora dentro de la fase de mejoramiento del proceso. La búsqueda tabú es una técnica iterativa que en cada paso se mueve de una solución factible s, de un problema de optimización combinatorio, hacia una solución s' que proporcione el mejor incremento (o decremento) de la función de costos f en $V^*$ que es un subconjunto de una vecindad de s denotada N(s).

Existen dos diferencias esenciales entre una técnica de mejoramiento local y el método antes mencionado. La primera es la posibilidad de poder moverse a una solución s' peor que la solución s, y la segunda diferencia es que el conjunto V* es el que guía la búsqueda tabú.

**Aspectos importantes en un método por búsqueda tabú**:

1. La búsqueda tabú hace *uso sistemático de la memoria* (en un computador), pues al aceptar una solución s' peor que la solución s, pueden ocurrir ciclos, por lo que se debe incorporar una estructura de memoria tal que prohiba o penalice ciertos movimientos que podrían hacer retornar a estados recién visitados. Aquí se originan los movimientos que se caracterizan como tabú. Esto conduce a una orientación general del conjunto V* que es estratégicamente generado y no tan aleatorio.

2. Las restricciones tabú no son inviolables bajo toda circunstancia. Cuando un movimiento tabú proporciona una solución mejor que cualquier otra encontrada, su condición tabú puede eliminarse, a esto se le denomina *criterio de aspiración*.

3. Un movimiento permanece tabú solo durante un cierto número de iteraciones, proporcionando una *estrategia de olvido*.

Por lo tanto, el criterio de aspiración y la restricción tabú, juegan un papel de guiar el proceso de búsqueda de la solución óptima.

## 3. PARTICIONAMIENTO USANDO BUSQUEDA TABU (METODO ORIGINAL)

En esta sección se presenta resumidamente el método original de particionamiento usando búsqueda tabú para obtener una partición de un conjunto $\Omega$ con n individuos, denotados por $x_i$ y ponderados con pesos positivos $w_i$ tales que $\sum_{i=1}^{n} w_i = 1$, a los que se les ha medido p variables cuantitativas $x^j$. Esta propuesta se ha basado en el algoritmo de transferencia [4, 7] de Régnier.

Se desea obtener una partición del conjunto $\Omega$ en k clases bien separadas entre ellas y lo más homogéneas posibles. Sea P = ($P_1$,...,$P_k$) una partición del conjunto $\Omega$ en k clases; para obtener las características antes mencionadas se minimizará la inercia intra-clase:

$$W(P) = \sum_{j=1}^{k} I(P_j) \, ,$$

donde

$$I(P_j) = \sum_{x_i \in P_j} w_i \, \| x_i - g(P_j) \|^2 \, ,$$

siendo $g(P_j)$ el centro de gravedad de la clase $P_j$, esto es

$$g(P_j) = \frac{1}{\mu_j} \sum_{x_i \in P_j} w_i x_i \, ,$$

con

$$\mu_j = \sum_{x_i \in P_j} w_i \, .$$

Para modelar el problema de particionamiento se usará un vector de tamaño n y en cada entrada de este vector se asignará un número del 1 al k, representando la clase a la que pertenece ese individuo.

La inicialización del método se hace en forma aleatoria (utilizamos un algoritmo aditivo para la generación de números aleatorios: ver [Piza-Trejos-Murillo (1994)], esto es, se asigna aleatoriamente un número del 1 al k en cada entrada del vector que representa la partición P.

Se entenderá como vecindario de una partición P al conjunto de las particiones candidatas que se obtienen de esta al hacer un movimiento (transferir a un individuo $x_i$ de una clase a otra clase) de algún individuo. Se denotará N(P) el conjunto de todas las particiones candidatas a partir de la partición P.

Una partición es tabú si ha sido considerada en las últimas iteraciones para estudiar su vecindario. Las últimas particiones forman la lista tabú y la longitud de esta es un parámetro del método. Si es demasiado pequeño, el ciclado ocurrirá, pero si es demasiado grande, restringirá bastante la búsqueda para poder saltar los llamados "valles profundos" correspondientes a posibles óptimos locales [De los Cobos (1994). En la lista tabú lo que se almacena es la inercia intra-clase, pues la clasificación representada por (2, 1, 1, 1, 1) es la misma clasificación representada por (1, 2, 2, 2, 2) y es muy poco probable que, para datos reales, dos clasificaciones diferentes tengan la misma inercia intra-clase.

Una partición de N(P) es *admisible* si no es tabú o si el criterio de aspiración elimina su estatus de tabú. V* será el conjunto de todas las particiones admisibles de N(P). En cada iteración se escoge la mejor partición de P' de V* y la estructura de datos tabú se actualiza con la partición P, aunque la inercia intra-clase de P' sea peor que la inercia intra-clase de P. Con esto se busca trascender la optimalidad local.

Luego el método realiza iteraciones hasta que un número predeterminado de iteraciones sea superado.

**Algoritmo original**

**Inicio**

Leer número máximo de iteraciones
Determinar aleatoriamente la partición inicial
Mejor partición ← partición inicial
Partición anterior ← partición inicial
Número de iteración ← 0
**Repita**
Incremente el número de iteración
Crear el vecindario de la partición anterior
Calcular la inercia intra-clase de cada uno del vecindario
Escoger la mejor partición admisible
Actualice lista tabú con la inercia de la mejor partición admisible
Partición anterior ← mejor partición admisible
**Si** Inercia mejor partición admisible < Inercia mejor partición
**Entonces** Mejor partición ← mejor partición admisible
**Hasta** número de iteración = número máximo de iteraciones

**Fin**.

**Ejemplo de una ejecución del algoritmo original**

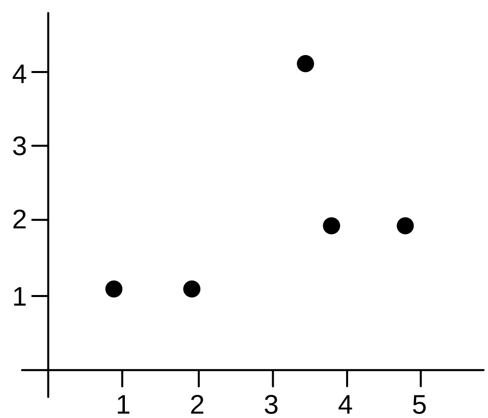


**Figura 1**. Gráfico de los puntos a clasificar.

El siguiente ejemplo es un grupo de datos que se construyó con el propósito de examinar el comportamiento del algoritmo ante un resultado conocido. La partición inicial en este caso no fue aleatoria para efectos didácticos de comprender el comportamiento del algoritmo, porque cuando se ejecutaba el algoritmo con soluciones aleatorias, se obtuvo la inercia óptima en la primera o segunda iteración.

Los datos son puntos en $R^2$, estos son: (1,1), (2,1), (4,2), (5,2) y (4,4), los cuales se pueden observar en la Figura 1.

La longitud de la lista tabú es de 5 y el número máximo de iteraciones fue de 10, pero solo se presentan las cuatro primeras iteraciones. Se desean encontrar tres clases que minimicen la inercia intra-clase. En la Tabla 1 se presenta la primera iteración de este algoritmo.

El número de iteración se encuentra en la primera columna de la Tabla 1. Los vectores que están en la segunda columna forman el vecindario de la partición inicial, denotado por N(1,1,1,1,1). En la tercera columna está la inercia de la partición correspondiente de la segunda columna. La cuarta columna es para designar si una partición es tabú y el $*$ señala la mejor partición admisible.

| Iteración | Clasificación | Inercia | Es tabú | Mejor admisible |
|---|---|---|---|---|
| 0 | 1 1 1 1 1 | 3.36 | | |
| 1 | 2 1 1 1 1 | 1.90 | | |
| | 3 1 1 1 1 | 1.90 | | * |
| | 1 2 1 1 1 | 2.75 | | |
| | 1 3 1 1 1 | 2.75 | | |
| | 1 1 2 1 1 | 3.20 | | |
| | 1 1 3 1 1 | 3.20 | | |
| | 1 1 1 2 1 | 2.55 | | |
| | 1 1 1 3 1 | 2.55 | | |
| | 1 1 1 1 2 | 2.20 | | |

**Tabla 1**. Primera iteración del algoritmo original.

Se observa en la Tabla 1 que existen dos particiones admisibles que nos proporcionan los mejores valores de la inercia, por lo que, se puede escoger cualquiera de los dos, en este caso se elige el segundo. En la Tabla 2 está la segunda iteración del algoritmo.

**Tabla 2**. Segunda iteración del algoritmo original.

| Iteración | Clasificación | Inercia | Es tabú | Mejor admisible |
|---|---|---|---|---|
| 1 | 3 1 1 1 1 | 1.90 | | |
| 2 | 1 1 1 1 1 | 3.36 | Sí | |
| | 2 1 1 1 1 | 1.90 | Sí | |
| | 3 2 1 1 1 | 0.67 | | * |
| | 3 3 1 1 1 | 0.77 | | |
| | 3 1 2 1 1 | 1.87 | | |
| | 3 1 3 1 1 | 2.87 | | |
| | 3 1 1 2 1 | 1.47 | | |
| | 3 1 1 3 1 | 3.17 | | |
| | 3 1 1 1 2 | 1.07 | | |
| | 3 1 1 1 3 | 2.87 | | |

En la segunda iteración hay dos particiones tabú, como se observa en la Tabla 2, pues uno es equivalente a la partición inicial y el otro a la partición anterior, sin embargo, existe otra partición que mejora bastante la inercia intra-clase.

En la Tabla 3 se puede observar la diferencia del algoritmo original con un algoritmo descendente, pues acepta como siguiente partición una cuya inercia intra-clase es peor que la anterior, pero esto le sirve para salir de un mínimo local y continuar el proceso de búsqueda por otra región.

Ya en la cuarta iteración se obtiene el óptimo global, pues en este caso era conocido desde el inicio, el cual se señala con tres asteriscos. La clasificación óptima que se obtuvo es:

**Tabla 3**. Tercera iteración del algoritmo original.

| Iteración | Clasificación | Inercia | Es tabú | Mejor admisible |
|---|---|---|---|---|
| 2 | 3 2 1 1 1 | 0.67 | | |
| 3 | 1 2 1 1 1 | 2.75 | | |
| | 2 2 1 1 1 | 0.77 | | |
| | 3 1 1 1 1 | 1.90 | Sí | |
| | 3 3 1 1 1 | 0.77 | | * |
| | 3 2 2 1 1 | 1.00 | | |
| | 3 2 3 1 1 | 1.50 | | |
| | 3 2 1 2 1 | 1.40 | | |
| | 3 2 1 3 1 | 2.10 | | |
| | 3 2 1 1 2 | 1.40 | | |
| | 3 2 1 1 3 | 1.90 | Sí | |

{(4,2), (5,2)}, {(4,4)} y {(1,1), (2,1)} con una inercia intra-clase de 0.2.

En la práctica se desconoce el óptimo global, así que se debe ejecutar el algoritmo una mayor cantidad de iteraciones, con diversas soluciones iniciales y escoger la mejor obtenida en todos los intentos.

**Tabla 4**. Cuarta iteración del algoritmo original.

| Iteración | Clasificación | Inercia | Es tabú | Mejor admisible |
|---|---|---|---|---|
| 3 | 3 3 1 1 1 | 0.77 | | |
| 4 | 1 3 1 1 1 | 2.75 | | |
| | 2 3 1 1 1 | 0.67 | Sí | |
| | 3 1 1 1 1 | 1.90 | Sí | |
| | 3 2 1 1 1 | 0.67 | Sí | |
| | 3 3 2 1 1 | 0.60 | | |
| | 3 3 3 1 1 | 1.57 | | |
| | 3 3 1 2 1 | 0.50 | | |
| | 3 3 1 3 1 | 2.27 | | |
| | 3 3 1 1 2 | 0.20 | | *** |
| | 3 3 1 1 3 | 2.23 | | |
| | | | | |

## 4. PARTICIONAMIENTO USANDO BUSQUEDA TABU (METODO MEJORADO)

En esta sección se presenta el método mejorado de particionamiento usando búsqueda tabú para obtener una partición del conjunto $\Omega$ con n individuos.

Se observa que un movimiento dentro de la búsqueda tabú consiste en transferir a un individuo de una clase a otra. Se P la partición de $\Omega$ antes del movimiento y sea P' la partición de $\Omega$ después del movimiento. La variación de la inercia intra-clase es:

$\Delta W = W(P') - W(P)$.

Para simplificar el cálculo de la nueva inercia de P´ sin hacer todos los cálculos se usarán las siguientes fórmulas [10].

Notación:

- Sea $P_j$ la clase donde está el individuo $x_i$ que se va a transferir.
- Sea $P_l$ la clase que fue escogida para transferir al individuo $x_i$.

Entonces:

$\Delta W = I(P_j - \{x_i\}) + I(P_1 \cup \{x_i\}) - I(P_j) - I(P_1)$.

Sin embargo, como se demuestra en [10] no es necesario calcular los centros de gravedad de $P_j - \{x_i\}$ ni el de $P_l \cup \{x_i\}$, sólo se llevarán a cabo en el caso de ser escogidos, entonces si los pesos son iguales se tiene:

- si $|P_j| > 1$, entonces

$$\Delta W = \frac{|P_1|}{n(|P_1|+1)} \| g(P_1) - x_i \|^2 - \frac{|P_j|}{n(|P_j|-1)} \| g(P_1) - xi \|^2$$

- si $|P_j| = 1$, entonces

$$\Delta W = \frac{|P_1|}{n(|P_1|+1)} \| g(P_1) - x_i \|^2 .$$

Luego, si la partición es escogida, los centros de gravedad se calculan de la siguiente forma:

$$g(P_1 \cup \{x_i\}) = \frac{1}{|P_1|+1}\left(|P_1| g(P_1) + x_i\right),$$

- si $|P_j| > 1$, entonces

$$g(P_j - \{x_i\} = \frac{1}{|P_j|-1}\left(|P_j| g(P_j) - x_i\right),$$

- si $|P_j| = 1$, entonces

$$g(P_j - \{x_i\}) = 0.$$

Además de estas fórmulas, existen fórmulas análogas para cuando los pesos de los individuos no son iguales. [Piza-Trejos-Murillo (1994)]

Para el nuevo modelo del problema de particionamiento se usará un vector de tamaño n y en cada entrada de este vector un número del 1 al k representando la clase a la que pertenece ese individuo. La inicialización del método se hace en forma aleatoria [Piza-Trejos-Murillo (1994)], esto es, se asigna aleatoriamente un número del 1 al k en cada entrada del vector que representa la partición P.

Se entenderá como vecindario de una partición P, las particiones candidatas que se obtienen de esta al hacer un movimiento (transferir a un individuo $x_i$ de una clase a otra clase) de algún individuo. Se denotará N(P) el conjunto de todas las particiones candidatas a partir de la partición P.

Una partición es tabú si una de sus clases está prohibida en la lista tabú, esto es, si esta clase ha sido considerada en las últimas particiones. Lo que se pone en la lista tabú es una representación de la clase, en la partición $P_l$ que contiene al individuo transferido $x_i$. De esta forma se evita el ciclado, pues al no aceptar una partición que contiene la clase prohibida de una partición ya analizada, no es posible llegar a una partición cuya vecindad ya fue estudiada. Las últimas particiones forman la lista tabú y la longitud de esta es un parámetro del método, si es demasiado pequeña el ciclado ocurrirá, pero si es demasiado grande, restringirá bastante la búsqueda para poder saltar los llamados "valles profundos" correspondientes a posibles óptimos locales [De los Cobos (1994)].

Una partición de N(P) es *admisible* si no es tabú o si el criterio de aspiración elimina su estatus de tabú. V* será el conjunto de todas las particiones admisibles de N(P). En cada iteración se escoge la mejor partición de P' de V* y la estructura de datos tabú se actualiza con la partición P, aunque la inercia intra-clase de P' sea peor que la inercia intra-clase de P. Con esto se busca trascender la optimalidad local. Luego el método realiza iteraciones hasta que un número predeterminado de iteraciones sea superado.

### Nuevo algoritmo propuesto

**Inicio**

Leer número máximo de iteraciones

Determinar aleatoriamente la partición inicial

Mejor partición ← partición inicial

Partición anterior ← partición inicial

Número de iteración ← 0

**Repita**

Incremente el número de iteración

Crear el vecindario de la partición anterior

Modificar la inercia intra-clase (usando $\Delta W$) de cada uno del vecindario

Escoger la mejor partición admisible

Actualice lista tabú con la clase que contiene el individuo transferido

Modifique los centros de gravedad de la mejor partición admisible

Partición anterior ← mejor partición admisible

**Si** Inercia mejor partición admisible < Inercia mejor partición

**Entonces** Mejor partición ← mejor partición admisible

**Hasta** número de iteración = número máximo de iteraciones

**Fin**.

### Ejemplo de una ejecución del nuevo algoritmo propuesto

Considere el ejemplo de la Figura 1. La longitud de la lista tabú es de 5 y el número máximo de iteraciones fue de 10, pero solo se presentan las tres primeras iteraciones. Se desean encontrar tres clases que minimicen la inercia intra-clase. La partición inicial está dada por (3, 1, 1, 1, 1), por lo tanto la lista tabú inicial es:

Lista tabú: 10000 → 00000 → 00000 → 00000 → 00000.

Esto es, se prohibe que el primer individuo esté en una clase unitaria. Luego la primera iteración está dada en la Tabla 5
.

**Tabla 5**. Primera iteración del nuevo algoritmo propuesto

| **Iteración** | **Clasificación** | **Inercia** | **Es tabú** | **Mejor admisible** |
|---|---|---|---|---|
| 0 | 3 1 1 1 1 | 1.90 | | |
| 1 | 1 1 1 1 1 | 3.36 | | |
| | 2 11 1 1 | 1.90 | Sí | |
| | 3 2 1 1 1 | 0.67 | Sí | * |
| | 3 3 1 1 1 | 0.77 | | |
| | 3 1 2 1 1 | 1.87 | Sí | |
| | 3 1 3 1 1 | 2.87 | | |
| | 3 1 1 2 1 | 1.47 | Sí | |
| | 3 1 1 3 1 | 3.17 | | |
| | 3 1 1 1 2 | 1.07 | Sí | |
| | 3 11 1 3 | 2.87 | | |

Se observa en la Tabla 5 que la mejor partición no tabú es: (3, 3, 1, 1, 1), pero existe otra partición con inercia intra-clase que mejora la mejor encontrada hasta el momento, esto es se satisface el criterio de aspiración, en este caso se elige la partición (3, 2, 1, 1, 1) y la lista tabú sería:

Lista tabú: 10000→01000→00000→00000→ 00000.

La segunda iteración del nuevo algoritmo propuesto está dada en la Tabla 6. En esta iteración hay dos particiones no tabú con inercia intra-clase menor, se escoge cualquiera.

.

**Tabla 6**. Segunda iteración del nuevo algoritmo propuesto.

| **Iteración** | **Clasificación** | **Inercia** | **Es tabú** | **Mejor admisible** |
|---|---|---|---|---|
| 1 | 3 2 1 1 1 | 0.67 | Sí | |
| 2 | 1 2 1 1 1 | 2.75 | Sí | |
| | 2 21 1 1 | 0.77 | | * |
| | 3 1 1 1 1 | 1.90 | Sí | |
| | 3 3 1 1 1 | 0.77 | | |
| | 3 2 2 1 1 | 1.00 | Sí | |
| | 3 2 3 1 1 | 1.50 | Sí | |
| | 3 2 1 2 1 | 1.40 | Sí | |
| | 3 2 1 3 1 | 2.10 | Sí | |
| | 3 2 1 1 2 | 1.40 | Sí | |
| | 3 21 1 3 | 1.90 | Sí | |

En la Tabla 6 se puede observar la diferencia del algoritmo propuesto con un algoritmo descendente, pues acepta como siguiente partición uno cuya inercia intra-clase es peor que la anterior, pero esto le sirve para salir de un mínimo local y continuar el proceso de búsqueda por otra región y la lista tabú sería:

Lista tabú: 10000 $\rightarrow$ 01000 $\rightarrow$ 11000 $\rightarrow$ 00000 $\rightarrow$ 00000.

**Tabla 7**. Tercera iteración del nuevo algoritmo propuesto.

| **Iteración** | **Clasificación** | **Inercia** | **Es tabú** | **Mejor admisible** |
|---|---|---|---|---|
| 2 | 2 2 1 1 1 | 0.77 | | |
| 3 | 1 2 1 1 1 | 2.75 | Sí | |
| | 3 21 1 1 | 0.67 | Sí | |
| | 2 1 1 1 1 | 1.90 | Sí | |
| | 2 3 1 1 1 | 0.67 | Sí | |
| | 2 2 2 1 1 | 1.57 | | |
| | 2 2 3 1 1 | 0.60 | Sí | |
| | 2 2 1 2 1 | 2.27 | | |
| | 2 2 1 3 1 | 0.50 | Sí | |
| | 2 2 1 1 2 | 2.23 | Sí | |
| | 2 21 1 3 | 0.20 | Sí | *** |

Ya en esta iteración se obtiene el óptimo global, el cual se señala con tres asteriscos. La clasificación óptima que se obtuvo es: {(4,2), (5,2)], [(1,1), (2,1)} y {(4,4)} con una inercia intra-clase de 0.2.

En la práctica se desconoce el óptimo global, así que se debe ejecutar el algoritmo una mayor cantidad de iteraciones. En la actualidad se está trabajando sobre una técnica de diversificación a largo plazo de la búsqueda, esto es, intensificar la búsqueda por las regiones que no se han analizado.

## 5. RESULTADOS NUMERICOS

Los dos algoritmos de las secciones anteriores se han ejecutado con varios juegos de datos muy utilizados en la literatura consultada, entre los cuales se tienen: las notas escolares [11] (matriz de $9 \times 5$), los peces de Amiard [Cailliez-Pagés (1976)] (matriz de $24 \times 16$), los Iris de Fisher [Everitt (1993)] (matriz de $150 \times 4$), y la sociomatriz de Thomas [Cailliez-Pagés (1976)] (matriz de $24 \times 24$).

Estos algoritmos se ejecutaron en un computador personal Pentium de 166 MHz y los resultados están en las siguientes tablas. El contador de tiempo que se utiliza mide hasta 0.05 segundos.

En las notas escolares la longitud de la lista tabú es de 10 y el número máximo de iteraciones es de 20. En los peces de Amiard la longitud de la lista tabú es de 20 y el número máximo de iteraciones es de 30. En los Iris de Fisher la longitud de la lista tabú es de 25 y el número máximo de iteraciones es de 75. En la sociomatriz de Thomas la longitud de la lista tabú es de 20 y el número máximo de iteraciones es de 40.

**Tabla 8**. Resultados de las corridas de los dos métodos en dos clases.

| **2 CLASES** | **METODO ORIGINAL** | | **METODO MEJORADO** | |
|---|---|---|---|---|
| **3 CLASES** | **METODO ORIGINAL** | | **METODO MEJORADO** | |
| **Ejemplos**: | Inercia | Tiempo | Inercia | Tiempo |
| Notas escolares | 16.81 | 0.16 seg | 16.81 | 0.00 seg |
| Peces de Amiard | 32213.38 | 2.86 seg | 32213.38 | 0.22 seg |
| Iris de Fisher | 0.52 | 153.63 seg | 0.52 | 5.05 seg |
| Sociomatriz de Thomas | 271.83 | 5.99 seg | 271.83 | 0.44 seg |

**Tabla 9**. Resultados de las corridas de los dos métodos en tres clases.

**Tabla 10**. Resultados de las corridas de los dos métodos en cuatro clases.

| **4 CLASES** | **METODO ORIGINAL** | | **METODO MEJORADO** | |
|---|---|---|---|---|
| **Ejemplos**: | Inercia | Tiempo | Inercia | Tiempo |
| Notas escolares | 10.47 | 0.22 seg | 10.47 | 0.05 seg |
| Peces de Amiard | 18281.39 | 4.34 seg | 18281.39 | 0.33 seg |
| Iris de Fisher | 0.37 | 231.73 seg | 0.37 | 7.31 seg |
| Sociomatriz de Thomas | 235.03 | 8.84 seg | 235.03 | 0.55 seg |

**Tabla 11**. Resultados de las corridas de los dos métodos en cinco clases.

| **5 CLASES** | **METODO ORIGINAL** | | **METODO MEJORADO** | |
|---|---|---|---|---|
| **Ejemplos**: | Inercia | Tiempo | Inercia | Tiempo |
| Notas escolares | 4.89 | 0.33 seg | 4.89 | 0.05 seg |
| Peces de Amiard | 14497.81 | 5.44 seg | 14497.81 | 0.49 seg |
| Iris de Fisher | 0.329 | 306.87 seg | 0.329 | 9.06 seg |
| Sociomatriz de Thomas | 202.58 | 12.30 seg | 202.58 | 0.82 seg |

## 6. CONCLUSION

En este trabajo se utilizó el arte de la optimización combinatoria basada en búsqueda tabú para tratar de escapar de la optimalidad local y guiar la búsqueda tratando de eliminar ciclos y aceptando algunas veces particiones peores para poder analizar otras regiones escapando de la atracción de los valles profundos.

En el primer método propuesto no es posible que se de el criterio de aspiración, pues si una partición es tabú, es porque ya fue analizado su vecindario, así que no es posible que la inercia intra-clase de esta partición sea mejor que todas las encontradas hasta el momento. Esto fue una de las modelaciones que se perfeccionó en el segundo método, el cual evita el ciclado dando estatus tabú a las particiones que tengan una clase tabú, esto es, una clase prohibida porque formó parte de un análisis previo en una partición escogida.

Además, es de gran importancia para ahorrar cálculos, que sólo se calculen los cambios que tiene la inercia intra-clase de una partición a la siguiente, así como, modificar los centros de gravedad a solo las particiones escogidas como mejores. Por estas razones, se recomienda que los vecindarios sean medianamente grandes, para analizar más particiones con mínimos cálculos.

La ejecución de este algoritmo depende de la cantidad de elementos a clasificar y del número de clases que se desean.

Los resultados obtenidos por el nuevo método propuesto, son realmente mejores que el algoritmo original en cuanto al tiempo real en el computador, pues tiene mejoras de 26 veces en promedio de los casos analizados.

## REFERENCIAS


CAILLIEZ, F. et J.P. PAGES (1976): "Introducción à l´Analyse des Données", SMASH. París.

DE LOS COBOS, S. (1994): "La técnica de la búsqueda tabú y sus aplicaciones". Tesis Doctoral. Universidad Nacional Autónoma de México.

DE LOS COBOS, S.; B.R: PEREZ y M.A. GUTIERREZ (1995): "Lineamientos de implantación de la búsqueda tabú para los problemas de calendarización", En: Memorias IX Simposio Métodos Matemáticos Aplicados a las Ciencias. Turrialba, Costa Rica, 23-32.

ESPINOZA, J.L. y J. TREJOS (1989): "Clasificación por particiones", **Revista Ciencia y Tecnología**, XIII(1-2), 129-154.

EVERITT, B.S. (1993): "Cluster Analysis", 3ª. edición. Edward Arnold, Londres.

GLOVER, F. and E. TAILLARD (1993): "Tabu search: an introduction", **Annals of Operations Research**, 41(1-3), 1-28.

LERMAN, I.C. (1981): "Classification et Analyse Ordinale des Données", Dunod, París.

MURILLO, A. (1996): "Particionamiento usando búsqueda tabú", en IV Encuentro Centroamericano de Investigadores en Matemática, Antigua Guatemala, febrero.

MURILLO-FERNANDEZ, A. et J. TREJOS ZELAYA (1996): "Classification tabou basée en transferts", Quatriemes Journees de la Societe Francophone de Classification, Avec le soutien de la Ville de Vannes et de l' Université de Bretagne Sud.

PIZA, E.; J. TREJOS and A. MURILLO (1994): "Clasificación Automática: particiones utilizando algoritmos genéticos y de sobrecalentamiento simulado", Primer informe del proyecto de investigación, 114-94-228, Universidad de Costa Rica, San Pedro.

SCHEKTMAN, Y. (1978): "Estadística descriptiva (análisis lineal de datos multidimensionales)", I parte. En: Memorias I Simposio Métodos Matemáticos Aplicados a las Ciencias, Universidad de Costa Rica, San Pedro, 9-67.