ESCUELA TÉCNICA SUPERIOR DE INGENIERÍA INFORMÁTICA Y

TELECOMUNICACIONES

PRACTICA 3.B:
ENFRIAMIENTO SIMULADO
BÚSQUEDA LOCAL REITERADA
EVOLUCIÓN DIFERENCIAL

Realizado por:
González Mairena, Juan Jesús
DNI: 77206717H
jiuxej@correo.ugr.es
3º CURSO INGENIERÍA INFORMÁTICA
GRUPO: 3 - JUEVES(17:30-19:30)

Contents

1 Descripción del problema **2**

2 Descripción de funciones generales aplicadas al problema **2**

3 Descripción funciones especificas para los nuevos algoritmos **5**

4 Descripción Enfriamiento Simulado **6**

5 Descripción Búsqueda Local Reiterada **8**

6 Descripción Evolución Diferencial **8**

7 Tablas de resultados **9**

7.1 Tabla de ejecuciones Enfriamiento Simulado 9
7.2 Tabla de ejecuciones Búsqueda Local Reiterado 10
7.3 Tabla de ejecuciones Evolucion Diferencial RAND 10
7.4 Tabla de ejecuciones Evolucion Diferencial BEST 10
7.5 Tabla de ejecuciones Relief 10
7.6 Tabla de ejecuciones BL 10
7.7 Tabla de ejecuciones AGG-BLX 11
7.8 Tabla de ejecuciones AM-[10 1] 11
7.9 Resultados Globales 11

8 Análisis de resultados **11**

8.1 Análisis Tasa de Clase 12
8.2 Análisis Tasa de Reducción 14
8.3 Análisis de Agregado 15
8.4 Análisis de tiempo 16

9 Cambio en Enfriamiento simulado **17**

10 Experimento **21**

1 Descripción del problema

El Aprendizaje de Pesos en Características (APC) trata sobre la optimización de un clasificador de un conjunto de objetos a partir del uso de un vector de pesos que usaremos para ponderar las características dichos objetos. La estructura que tendremos para la optimización seria:

- Un vector de vectores de double de tamaño MxN donde introduciremos los datos de los objetos.
- Un vector de pesos de tamaño N, que usaremos para la clasificación de los objetos.

Tanto los datos de la matriz como el vector de pesos debe estar entre [0,1], por lo que debemos de Normalizar los datos de la matriz antes de introducirlo en nuestro algoritmo.
Para la normalización usaremos la siguiente formula:

$$x_j{}^N = \frac{x_j - Min_j}{Max_j - Min_j}$$

Para comprobar la calidad del vector de pesos obtenido por los algoritmos usaremos la siguiente función que debemos de maximizar:

$$F(W) = \alpha * TasaClas + (1 - \alpha) * TasaRed$$

Donde $\alpha = 0.5$ y la Tasa de Reducción sera el porcentaje de características que descartamos en el vector de pesos y la Tasa Clasificación de la función el porcentaje de aciertos de etiquetas sobre el conjunto de test.

$$tasa\_class = \frac{n^o\_instancias\_bien\_clasificados\_en\_T}{n^o\_instancias\_en\_T}$$

$$tasa\_red = \frac{n^o\_valores\_wi < 0.2}{n^o\_caracteristicas}$$

Para determinar el conjunto de datos que usaremos como test dividiremos la matriz de datos en 5 secciones donde repartiremos equitativamente los datos y las etiquetas (5-fold cross validation). Realizaremos 5 ejecuciones del algoritmos usando en cada realización una partición como test. La función de para la partición la muestro más adelante.

2 Descripción de funciones generales aplicadas al problema

Como ya hemos dicho antes, nuestro problema consiste en clasificar objetos dadas una serie de características las cuales usaremos para comprobar la distancia de ese objeto frente al resto. Estas distancias entre características son ponderadas el vector de pesos W, teniendo en cuenta que si el valor de $w_i < 0.2$ esta no es tenida en cuenta para el calculo de la distancia. Para determinar la distancia usaremos:

$$d(e_1, e_2) = \sqrt{\sum_{i=0}^{N} (e_{1i} - e_{2i})^2}$$

Respecto a la partición la realizamos como hemos descrito anteriormente. La función es esta:

```
void particionMatriz:

    for i=0; i<5; i++
        Introduce vector double en matrizParticion
        Introduce vector double en targetParticion

    for i=0; i<fil; i++
        for j=0; j<col; j++
            Introduce dato i, j en matrizParticion en la posicion [i%5]

        Introduce dato i en targetParticion en la posicion [i%5]
```

También incluimos el pseudo-código de la normalización de los datos:

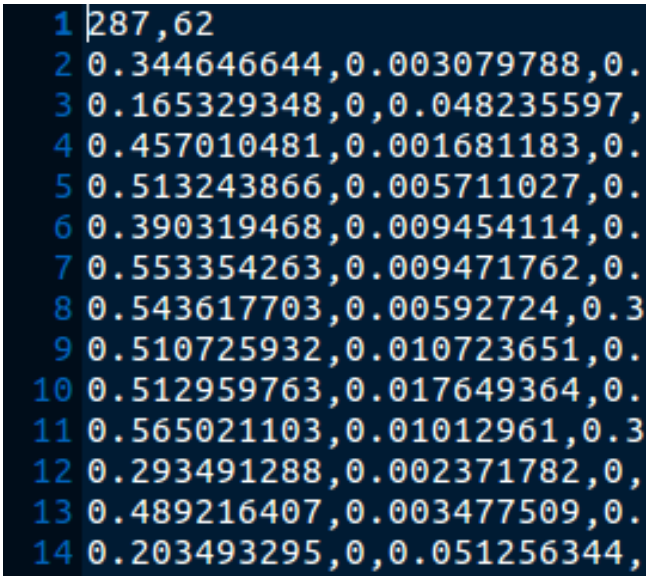
```
void normalizarDatos

    MAX = 0
    for i=0; i<fil; i++
        for j=0; j<col; j++
            Si dato en i,j es mayor que MAX
                MAX = dato i,j

    MIN = 9999
    for i=0; i<fil; i++
        for j=0; j<col; j++
            Si dato en i,j es menor que MIN
                MIN = dato i,j

    for i=0; i<fil; i++
        for j=0; j<col; j++
            Aplicamos formula de normalización sobre el dato i,j
```

Para la lectura de los archivos realizaremos una modificación sobre estos, pasándolos a un archivo .txt y estableciendo en la primera linea el numero de filas que contiene el archivo y el número de columnas. La estructura seria:



La función que usaremos para la lectura de los archivos sera:

```
void obtenerDatos

    Abrimos el fichero

    while No termine de leer el fichero
        Lee linea del fichero

        for j=0; j<col+1; j++
            while No se termine la columna
                Si lo leido no es una coma ni un salto de linea lo sumamos a la palabra
                Si lee un salto de linea añade la palabra a la matriz
                Si no añade la palabra al vector de etiqueta

    Cerramos el fichero
```

Para la función objetiva que debemos maximizar en este problema estara estructurado del siguiente modo:

```
double funcionEvaluacion

    Calculamos la Tasa de Clase respecto a la particion Test
    Calculamos la Tasa de Reduccion respecto al W obtenido
    Calculamos la calidad = TasaClase*0.5 + TasaReduccion*0.5

    Devolvemos calidad
```

Para el calculo de la tasa de Clase y Reducción usaremos:

```
double TasaClaseTest:

    Recorremos la particion que hemos establecido como test
        Recorremos las 5 particiones
            Si la particion es diferente a la que estamos probando realizamos lo siguiente
                Recorremos la particion selecionada elemento por elemento
                    Comparamos el elemento seleccionado en test con los demas elementos de otras particiones columna por columna
                        Si dicha columna tiene una ponderacion mayor a 0.2
                            Calculamos la distancia
                        Si la distancia es la mas pequeña encontrada
                            Establecemos como distancia minima

        Si hemos acertado la etiqueta del objeto
            Aumentamos el numero de aciertos

    Devolvemos (100.0 * Numero de aciertos) / Tamaño de la particion de Test
```

```
double TasaReduccion:

    for int i=0; i<numCaracteristicas; i++
        Contamos cuantos valores son menores o iguales que 0.2

    Devolvemos (100.0 * Numero de Valores menores o iguales que 0.2) / Numero de Caracteristicas total de W
```

3 Descripción funciones especificas para los nuevos algoritmos

Para esta practica resolveremos el problema APC haciendo uso de Enfriamiento Simulado (ES), Búsqueda Local Reiterada(ILS) y Evolución Diferencial(ED). El único algoritmo que usara parte del código anteriormente usado sera ED ya que necesitaremos crear una población de cromosomas:

```
struct Cromosoma
    vector de pesos W
    Calidad de W
```

Ademas de necesitar inicializarlo correctamente:

```
vector<Cromosoma> InicializarPoblacion:

    Inicializamos la poblacion con tam vectores de pesos
    for i=0; i<tam; i++
        Hacemos uso de la funcion Normal(0, 0.3) para los pesos del vector que estamos creando
            Si algun peso es negativo establecemos ese peso como 0

        Añadimos el vector de pesos a la poblacion junto con la calidad que genera
    Devolvemos la poblacion creada
```

Por otra parte ILS usaran la búsquedas local para mejorar los vecinos obtenidos con las mutaciones. El pseudo-codigo seria el siguiente:

```
void BusquedaLocal

    Mientras no se alcance el maximo de Evaluaciones
        Si se han usado todos los indices posibles meclamos el vector de indices

        Seleccionamos el correspondiente indice del vector de indices
        Obtenemos el vecino del vector w sumandole a la posicion indice de w lo obtenido en Normal(0,0.4)

        Evaluamos la calidad del vecino

        Si es mejor la calidad actual
            Sustituimos calidad actual y conservamos el vecino
        Si no recuperamos el vector anterior
```

4 Descripción Enfriamiento Simulado

El Enfriamiento Simulado(ES) es destacado por combinar explotación y exploración a medida que avanza el tiempo en su ejecución. Comenzando su funcionamiento aceptando vecinos peores que nuestra solución actual, aumentando así la posibilidad de no caer en óptimos locales (exploración) y terminando su funcionamiento simular a como se ejecutaría una búsqueda local (explotación). Para ello, debemos tener en cuenta la reducción de lo que llamaremos temperatura (tempK) que establecerá la probabilidad de que una solución vecina peor sea elegida o no para sustituir a la solución actual.

El pseudo-código de este algoritmo seria el siguiente:

```
void EnfriamientoSimulado

    Inicializamos el vector W inicial de forma aleatoria
    Establecemos dicho w como el mejor

    Calculamos su calidad y establecemos esta calidad como la mejor

    Iniciamos Temperatura Inicial y la Temperatura Kusando la formula [ (0.3*mejorCalidad)/(-log(0.3)) ]
    Establecemos cual sera la temperatura final a 0.001
    Inicializamos beta con temp0 y tempF

    Hasta que TempK < TempF, el numero de exitos nos sea 0 o no se alcance el numero maximo de evaluaciones.

        Mientras en numero de exitos sea menor al maximo y el numero de vecinos generados sea menor que el maximo

            Obtenemos un valor aleatorio del vector w
            Generamos un vecino de w

            Evaluamos la calidad del vecino

            Si la calidad es mayor o si este es elegido al azar haciendo uso de la formula exp((calidadVecino - calidad - 0.5)/(tempK))
                Cambiamos nuestro w por ese vecino
                Guardando siempre el que mejor calidad tenga


            Si el vecino no ha sido seleccionado recuperamos el w anterior


        Disminuimos la temperatura K

    Devolvemos el mejor w conseguido
```

Las temperaturas estarán iniciadas por las siguientes formulas o valores:

$$T_0 = \frac{\mu C(S_0)}{-\ln(\phi)}$$
$$T_f = 0.001$$

La disminución de la temperatura K sera dada por el criterio de Cauchy modificado:

$$T_{k+1} = \frac{T_k}{1 + \beta T_k} \qquad\qquad \beta = \frac{T_0 - T_f}{M T_0 T_f}$$

El criterio de parada principal para este algoritmo consistirá en conseguir $T_k < T_f$, aunque podría no ocurrir debido a otros criterios impuestos por nosotros.

5 Descripción Búsqueda Local Reiterada

La búsqueda local Reiterada también destaca por su equilibrio de exploración y explotación ya que emplea la obtención de soluciones mediante mutaciones una solución optima, los que nos permite explorar el espacio de búsqueda y esto unirlo a la búsqueda local para encontrar las mejores soluciones del entorno de dicho vecino mutado. El algoritmo tendrá el siguiente pseudo-código:

```
void BusquedaLocalReiterada

    Inicializamos el vector inical de forma aleatoria

    Realizamos la busqueda local de este vector inicial
    Establecemos la calidad del vector inicial como la mejor

    Realizamos n iteraciones mas
        Creamos el  vecino de w realizando las mutaciones usando Normal(0,0.4)

        Aplicamos la busqueda local sobre el vecino
        Obtenemos la calidad de este

        Si la calidad mejora, establecemos dicho vecino como el mejor w

    Devolvemos el mejor w
```

El numero de mutaciones y de vecinos mutados sera definido al inicio del algoritmo.

6 Descripción Evolución Diferencial

El siguiente tipo de algoritmo se basa en los algoritmos genéticos y la forma en la estos mutan. Estos algoritmos se centraran en mutar los genes de los cromosomas elegidos con una probabilidad en cada generación, seleccionando al azar un numero de padres o en otro caso haciendo uso del mejor de la población. Estos algoritmos no resulta ser buenos exploradores ya que siempre se mantienen el entorno que puede generar los padres que generan la mutación. Dividiremos por tanto el estudio en dos subtipos:

- **Evolución Diferencial RAND**: Se seleccionara 3 padres de forma aleatoria. Se mutara los genes del descendiente a partir de la siguiente formula:

$$V_{i,G} = X_{r1,G} + F(X_{r2,G} - X_{r3,G})$$

  Siendo $V_{i,G}$ el gen del hijo, F sera 0.5 y $X_{rj,G}$ los padres que usaremos para mutar.

- **Evolución Diferencial BEST**: Se seleccionara 2 padres de forma aleatoria. Se mutara los genes del descendiente combinando estos padres con el mejor de la población actual y el gen del cromosoma que competirá con el descendiente por quedarse en la población. Usaremos la siguiente formula:

$$V_{i,G} = X_{i,G} + F(X_{best,G} - X_{ri,G}) + F(X_{r1,G} - X_{r2,G})$$

Siendo $V_{i,G}$ el gen del hijo, F sera 0.5, $X_{rj,G}$ los padres que usaremos para mutar, $X_{best,G}$ representa al mejor de la población y $X_{i,G}$ representa al cromosoma que competirá con dicho descendiente.

El pseudo-código sera general para ambos, diferenciándose con una variable opción el siguiente:

```
void EvolucionDiferencial

    Iniciamos la poblacion de cromosomas inicial
    En ED-BEST debemos guardar el mejor de la poblacion

    Mientras no se alcance el maximo de iteraciones
        Recorremos la poblacion
            Seleccionamos 2 o 3 padres de forma aleatoria  dependiendo del tipo de ED que usaremos

            Recorremos las caracteristicas para realizar las mutaciones
                Si el gen es seleccionado este es mutado con los valores de los padres
                Si el gen no es seleccionado este valor se mantiene igual que el cromosoam que hemos seleccionado en la poblacion

                Introducimos el valor en el nuevo descenciente
                Evaluamos el descendiente
        Introducimos el descendiente en la nueva poblacion.

        Sustituimos la poblacion anterior por la poblacion nueva
        En ED-BEST debemos actualizar el mejor de la poblacion

    Devolvemos el mejor de la poblacion
```

7 Tablas de resultados

A continuación muestro las tablas de los tiempos y el porcentaje de los resultados obtenidos por los algoritmos de esta practica y aquellos con los que compararemos:

7.1 Tabla de ejecuciones Enfriamiento Simulado

ES	Colposcopy				Ionosphere				Texture			
	%clas	%red	Agr.	T	%clas	%red	Agr.	T	%clas	%red	Agr.	T
P1	69,86	88,70	85,73	48,6823	86,78	85,29	91,23	25,129	87,72	85	88,40	75,9077
P2	67,24	85,48	84,98	48,5082	85,40	88,23	93,40	23,645	90,22	77,5	87,84	89,1259
P3	67,82	87,09	85,65	46,2856	85,40	91,17	92,73	25,139	90	80	89,09	90,9842
P4	73,91	80,64	78,04	52,0776	88,25	82,35	89,74	26,274	87,04	82,5	88,97	90,8072
P5	73,04	82,25	83,23	48,7621	82,91	88,23	92,68	23,044	88,86	85	89,31	95,0844
Media	70,38	84,83	83,52	48,8631	85,75	87,05	91,96	24,646	88,77	82	88,72	88,3818

7.2 Tabla de ejecuciones Búsqueda Local Reiterado

ILS	Colposcopy				Ionosphere				Texture			
	%clas	%red	Agr.	T	%clas	%red	Agr.	T	%clas	%red	Agr.	T
P1	71,17	91,93	89,93	121,465	84,28	91,17	91,36	77,3843	88,63	82,5	90,34	253,159
P2	69,43	91,93	90,79	124,141	85,76	91,17	94,87	74,4300	88,40	85	91,13	231,007
P3	67,82	91,93	90,70	118,259	87,18	94,11	94,20	72,2219	85	85	92,04	251,771
P4	72,60	91,93	90,70	118,626	86,47	85,29	91,21	84,5406	87,95	85	91,13	242,359
P5	73,91	88,70	89,96	121,428	83,27	91,17	94,87	74,3209	90,90	85	90,68	252,066
Media	70,99	91,29	90,42	120,783	85,39	90,58	93,30	76,5795	88,18	84,5	91,06	246,072

7.3 Tabla de ejecuciones Evolucion Diferencial RAND

ED-RAND	Colposcopy				Ionosphere				Texture			
	%clas	%red	Agr.	T	%clas	%red	Agr.	T	%clas	%red	Agr.	T
P1	69,86	79,03	80,03	214,889	85	91,17	94,17	88,0709	87,50	87,5	92,38	263,569
P2	71,61	67,74	76,97	221,457	88,25	91,17	94,87	82,5235	87,95	87,5	92,38	273,546
P3	74,78	77,41	80,81	218,560	90,39	91,17	94,87	85,2582	90,22	85	92,5	261,929
P4	70	83,87	81,40	221,230	88,96	91,17	94,15	85,9741	85,22	87,5	91,93	265,520
P5	69,56	72,58	78,39	223,381	83,62	91,17	94,87	86,0217	90,90	87,5	92,38	266,444
Media	71,16	76,12	79,52	219,903	87,24	91,17	94,59	85,5696	88,36	87	92,31	266,201

7.4 Tabla de ejecuciones Evolucion Diferencial BEST

ED-BEST	Colposcopy				Ionosphere				Texture			
	%clas	%red	Agr.	T	%clas	%red	Agr.	T	%clas	%red	Agr.	T
P1	72,48	70,96	76	148,879	90	79,41	88,29	106,653	90	80	88,18	305,746
P2	72,05	79,03	82,61	136,469	90,03	82,35	86,17	99,6579	93,63	67,5	81,93	384,682
P3	69,56	74,19	78,32	140,025	79,71	85,29	91,93	96,5233	91,13	75	85,22	313,701
P4	75,21	64,51	70,85	166,576	84,69	85,29	88,36	96,1802	90,45	80	87,72	291,48
P5	71,30	75,80	80	133,517	88,61	79,41	85,42	102,540	88,86	87,5	91,02	250,785
Media	72,12	72,90	77,56	145,093	86,61	82,35	88,03	100,310	90,81	78	86,81	309,278

7.5 Tabla de ejecuciones Relief

Relief	Colposcopy				Ionosphere				Texture			
	%clas	%red	Agr.	T	%clas	%red	Agr.	T	%clas	%red	Agr.	T
P1	72.93	58.06	66.96	0.13456	88.21	5.88	45.90	0.16114	94.09	17.5	55.57	0.41071
P2	72.93	90.32	78.78	0.09988	88.61	5.88	47.94	0.16099	94.32	10	49.54	0.43514
P3	72.61	59.68	63.17	0.13233	86.48	5.88	48.66	0.15196	94.54	10	53.18	0.43993
P4	73.91	54.84	59.88	0.13783	87.19	5.88	47.23	0.15225	93.41	10	51.81	0.44207
P5	71.74	70.97	68.82	0.1208	88.97	5.88	46.51	0.15685	92.50	10	51.81	0.43444
Media	72.82	66.77	67.52	0.12508	87.89	5.88	47.25	0.15664	93.77	11.5	46.10	0.43148

7.6 Tabla de ejecuciones BL

Relief	Colposcopy				Ionosphere				Texture			
	%clas	%red	Agr.	T	%clas	%red	Agr.	T	%clas	%red	Agr.	T
P1	71.18	91.94	89.07	15.7572	87.86	85.29	88.42	9.41049	80.68	80.00	85.45	17.9984
P2	68.99	87.10	85.79	13.6276	88.97	82.35	89.75	4.99247	89.32	81.50	87.61	17.8096
P3	74.35	85.48	84.85	9.6582	85.05	82.35	89.75	5.74484	87.05	85.00	91.14	21.5876
P4	71.30	85.48	83.09	11.6582	85.77	88.24	90.55	6.12548	91.14	82.50	90.34	39.3256
P5	70.87	87.10	88.29	20.628	85.41	76.47	86.10	7.15834	85.91	85.00	89.32	26.4397
Media	71.34	87.42	86.22	14.2683	86.61	82.94	88.91	6.68632	86.82	83.00	88.77	24.6322

7.7 Tabla de ejecuciones AGG-BLX

AGG-BLX	Colposcopy				Ionosphere				Texture			
	%clas	%red	Agr.	T	%clas	%red	Agr.	T	%clas	%red	Agr.	T
P1	68.12	93.55	84.71	66.8842	85.00	94.12	92.13	59.5769	81.82	90.00	87.73	197.929
P2	66.81	93.55	86.43	66.8186	80.43	91.18	92.73	66.19	83.41	87.50	88.30	185.83
P3	73.91	91.94	86.32	71.3577	78.29	91.18	91.30	64.5754	83.41	87.50	89.66	173.499
P4	63.48	95.16	87.05	67.9489	88.97	91.18	89.87	66.2464	80.23	90.00	89.09	172.207
P5	67.39	93.55	88.00	66.7256	77.22	94.12	90.63	59.0184	89.77	87.50	89.66	202.286
Media	67.94	93.55	86.50	67.947	81.98	92.35	91.33	63.1214	83.73	88.50	88.89	180.3502

7.8 Tabla de ejecuciones AM-[10 1]

AM-[10 1]	Colposcopy				Ionosphere				Texture			
	%clas	%red	Agr.	T	%clas	%red	Agr.	T	%clas	%red	Agr.	T
P1	71.61	83.87	84.18	90.3035	80.00	91.18	91.36	67.7682	92.03	80.00	88.64	248.486
P2	69.43	85.48	84.98	86.7142	87.19	95.71	93.45	68.6002	88.18	87.50	91.48	205.197
P3	69.13	87.10	84.78	88.186	77.22	94.12	92.06	60.6434	90.00	85.00	90.68	218.94
P4	72.61	85.48	82.21	89.1309	83.27	82.35	89.03	88.1315	87.05	85.00	88.86	218.632
P5	70.43	88.71	88.21	80.7379	83.27	88.24	92.69	71.9524	87.95	85.00	89.77	224.776
Media	71.64	86.13	84.87	87.0145	82.19	90.32	91.72	71.4191	89.09	84.50	89.89	223.2062

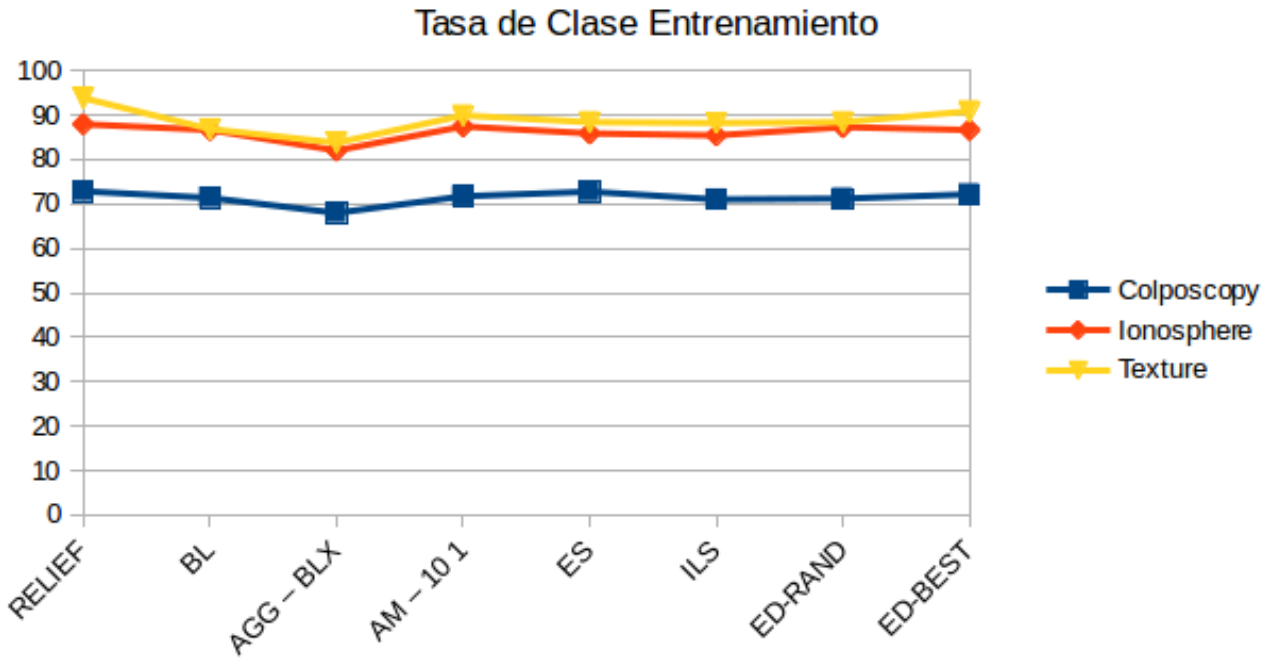
7.9 Resultados Globales

Media Global	Colposcopy				Ionosphere				Texture			
	%clas	%red	Agr.	T	%clas	%red	Agr.	T	%clas	%red	Agr.	T
Relief	72.82	66.77	67.52	0.12508	87.89	5.88	47.25	0.15664	93.77	11.5	46.10	0.43148
BL	71.34	87.42	86.22	14.2683	86.82	83.00	88.77	24.6322	86.82	83.00	88.77	24.6322
AGG-BLX	67.94	93.55	86.50	67.947	81.98	92.35	91.33	63.1214	83.73	88.50	88.89	180.3502
AM - [10 1]	71.64	86.13	84.87	87.0145	82.19	90.32	91.72	71.4191	89.09	84.50	89.89	223.2062
ES	72,73	80,64	80,38	34,4220	85,82	82,94	88,62	12,1947	88,36	82,50	90,38	50,49056
ILS	70,99	91,29	90,42	120,783	85,39	90,58	93,30	76,5795	88,18	84,50	91,06	246,0724
ED-RAND	71,16	76,12	79,52	219,903	87,24	91,17	94,59	85,5696	88,36	87	92,31	266,2016
ED-BEST	72,12	72,90	77,56	145,093	86,61	82,35	88,03	100,310	90,81	78	86,81	309,2788

8 Análisis de resultados

A continuación realizaremos un análisis de las tablas anteriores. Iremos columna a columna comparando los resultados obtenidos de todos los algoritmos conjuntos.

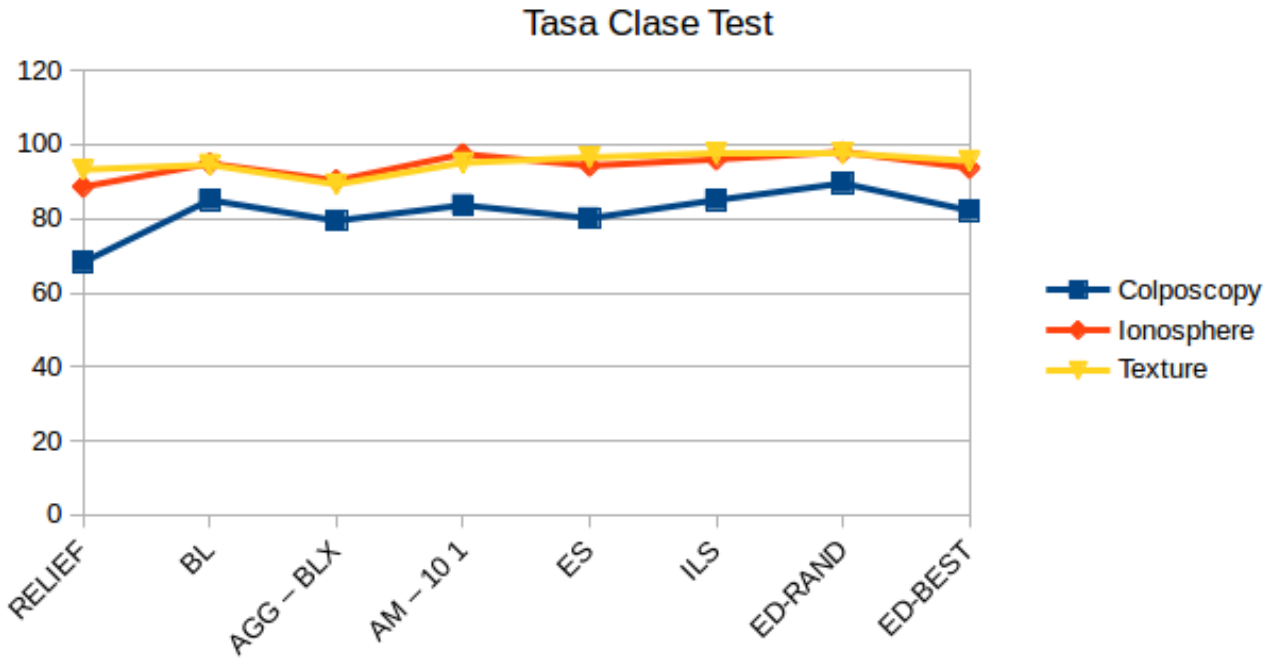
8.1 Análisis Tasa de Clase



La tasa de clase aquí mostrada muestra el numero de aciertos obtenidos respecto al conjunto de datos de entrenamiento. Observamos que el conjunto de datos que mejor resultado a dado en adaptar W en todos los algoritmos probados ha sido Texture, esto podría justificarse debido a la cantidad de ejemplos que tenemos. Respecto a los algoritmos utilizados, el mejor resultado lo ha dado Relief seguido de ED-BEST.

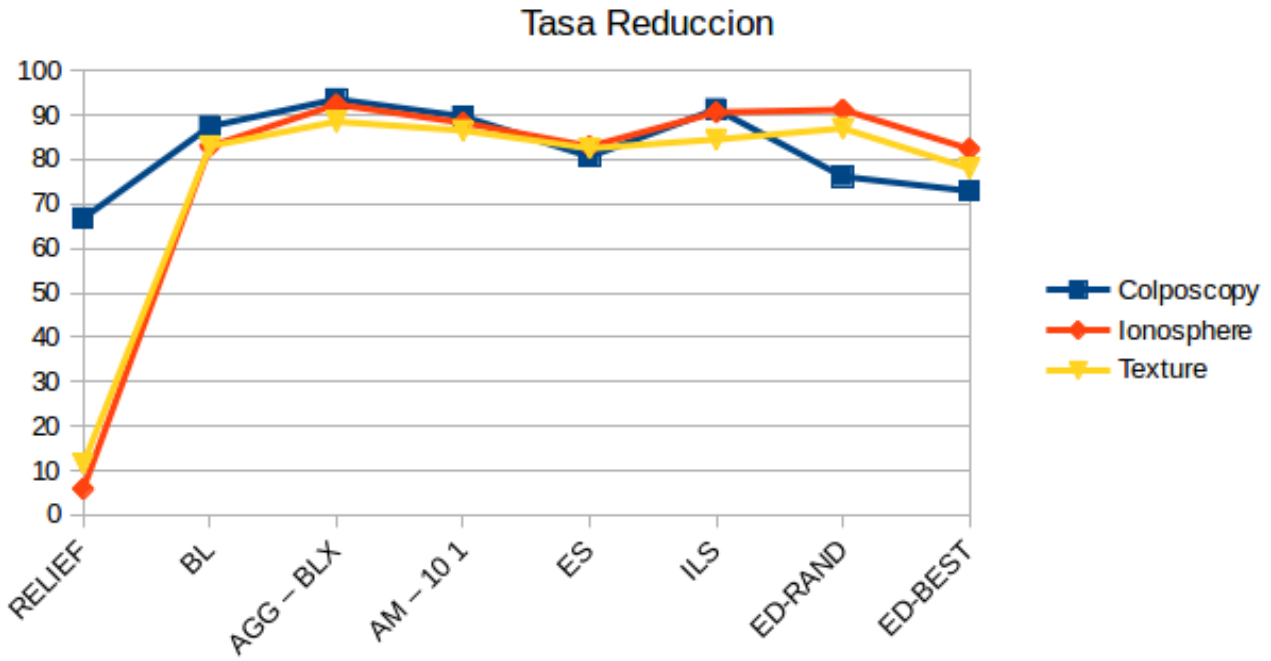
Respecto a los algoritmos añadidos en esta practica vemos que respecto a Tasa de clase todos han dado un valor muy similar, exceptuando ED-BEST con una pequeña mejora. Vemos que tanto ES como ILS han conseguido mejorar algo el valor de la tasa de clase comparado con BL, al igual que ED respecto a AGG-BLX y AM.

También mostrar la tasa de clasificación respecto al numero de acierto generado en la muestra de test.



Sobre esta gráfica podemos ver que los datos dados por los nuevos algoritmos son bastante buenos y similares siendo los mejores ILS y ED-RAND con un 97% en Texture, ademas de ver una mayor diferencia de mejora respecto a BL por parte de ES y ILS, y de la misma forma ED mejora o iguala el valor dado por AGG-BLX y AM.

8.2 Análisis Tasa de Reducción



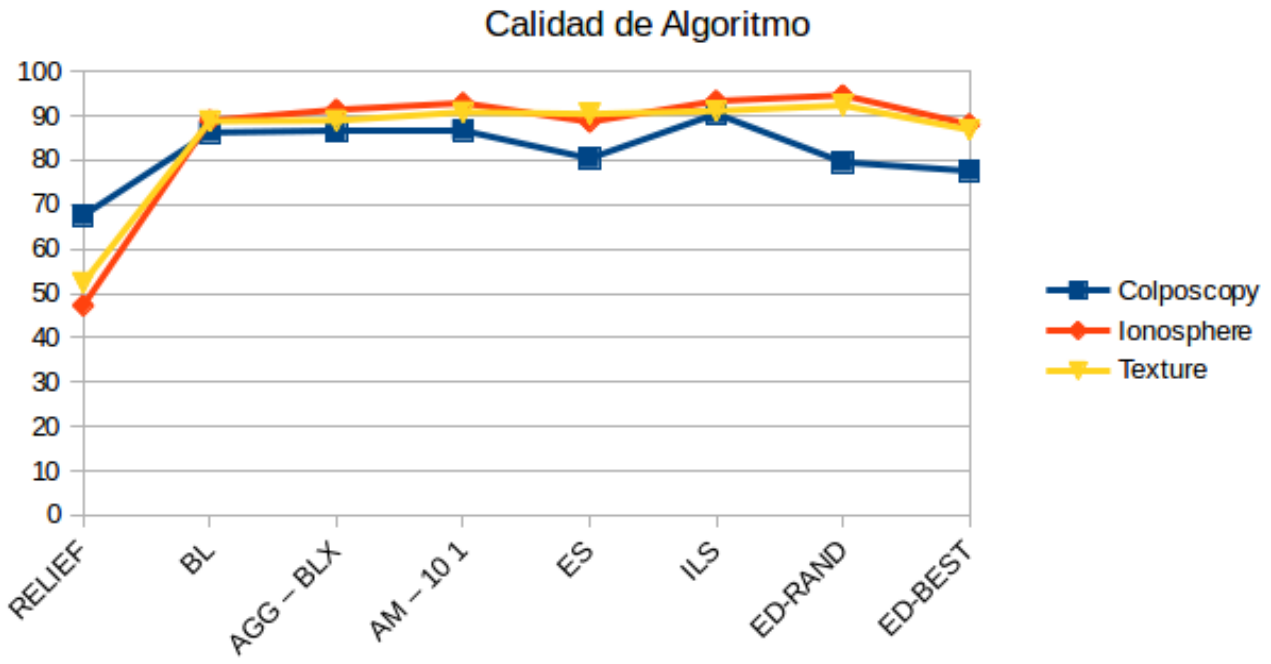
Observando la gráfica podemos ver diferencias entre los nuevos algoritmos sobre los diferentes conjuntos de datos. Vemos que el peor resultado estaría dado por ED-BEST en colposcopy y el mejor resultado en ED-RAND en Ionosphere.

Curiosamente podemos ver que ES consigue reducir de forma similar todos los conjuntos de datos dados, pero no ha conseguido mejorar los valores de BL, al contrario que ILS que ha mejorado de forma notable la reducción, sobre todo en Ionosphere, esto puede deberse a que aumenta la exploración que podria ofrecernos el BL simple ya que ILS hace uso del multi-arranque de BL.

Respecto a ED frente a AGG-BLX y AM los valores han sido mas bajos en ED-BEST y ED-RAND en Colposcopy, y algo similares en ED-RAND, pero sin conseguir mejora. Ademas podemos ver que la reducción ha sido mayor en RAND que en BEST esto podría deberse a que BEST se mueve entornos al mejor de la población, aumentando la intensidad, pero reduciendo la exploración, motivo por el cual puede estar dando peores valores que RAND.

De forma general, vemos que no se ha mejorado el valor dado por AGG-BLX, aunque esta esta seguido de forma muy próxima de ILS en Colposcopy.

8.3 Análisis de Agregado

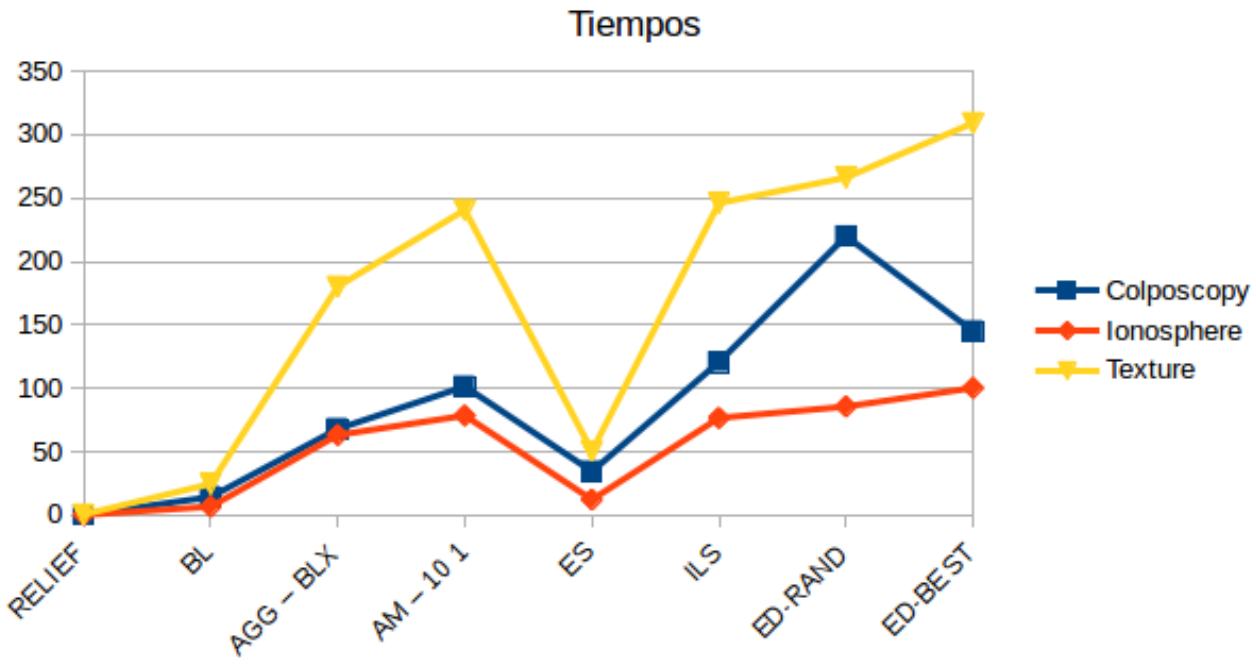


Analizando la gráficas de agregados vemos que el mejor resultado es dado por ED-RAND seguido de ILS, siendo este ultimo mejor en media sobre todos los conjuntos.

ES ha obtenido valores similares a BL o peores, esto se podría deber a que la disminucion del enfriamiento es demasiado rápida debido a la formula que estamos usando, por ello mas adelante realizaremos una mejora de este algoritmo.

Respecto a ED los resultados han sido un poco mejores en algunos conjuntos de datos por parte de ED-RAND frente a AGG y AM, siendo peores en ED-BEST por lo motivos antes argumentados.

8.4 Análisis de tiempo



Frente al tiempo podemos ver diferencias muy bruscas dadas sobre todo en ED-BEST. Este desproporcional aumento de tiempo sobre todo en Texture puede deberse a la gran cantidad de genes que debemos de recorrer para poder mutarlos.

Los tiempos son muy superiores a los visto hasta ahora exceptuando ES, que debido a su rápido enfriamiento los tiempos son menores a los dados en los demás algoritmos nuevos.

Respecto a la diferencia de tiempo de RAND con BEST podría deberse a que BEST realiza mas operaciones en su mutación que RAND.

9 Cambio en Enfriamiento simulado

Como hemos argumentado antes la formula usada para la disminucion de la temperatura en ES provoca que los resultados dados no alcancen lo esperado, ya que la temperatura se reduce demasiado rápido y reduciendo asi la capacidad de este algoritmo de exploración. Exponemos un ejemplo de como se reduce la temperatura con la formula usada hasta ahora:

```
Generados con temperatura K 12.4934: Exito = 62, Vecinos = 66
Generados con temperatura K 0.0239559: Exito = 21, Vecinos = 620
Generados con temperatura K 0.0119894: Exito = 3, Vecinos = 620
Generados con temperatura K 0.00799552: Exito = 2, Vecinos = 620
Generados con temperatura K 0.0059976: Exito = 1, Vecinos = 620
Generados con temperatura K 0.00479854: Exito = 0, Vecinos = 620
Total Vecinos Generados = 3166 de 620
Numero de Evaluaciones = 3166 de 15000
Temperatura K obtenida = 0.00399904
```

```
Generados con temperatura K 13.8574: Exito = 34, Vecinos = 36
Generados con temperatura K 0.0438639: Exito = 9, Vecinos = 340
Generados con temperatura K 0.0219667: Exito = 8, Vecinos = 340
Generados con temperatura K 0.0146522: Exito = 7, Vecinos = 340
Generados con temperatura K 0.0109921: Exito = 2, Vecinos = 340
Generados con temperatura K 0.00879505: Exito = 0, Vecinos = 340
Total Vecinos Generados = 1736 de 340
Numero de Evaluaciones = 1736 de 15000
Temperatura K obtenida = 0.00732998
```

```
Generados con temperatura K 14.5258: Exito = 40, Vecinos = 41
Generados con temperatura K 0.0369085: Exito = 23, Vecinos = 400
Generados con temperatura K 0.0184777: Exito = 4, Vecinos = 400
Generados con temperatura K 0.0123237: Exito = 0, Vecinos = 400
Total Vecinos Generados = 1241 de 400
Numero de Evaluaciones = 1241 de 15000
Temperatura K obtenida = 0.00924475
```

Vemos que el numero de éxitos es muy bajo debido al cambio brusco de la temperatura. Esto provoca que este algoritmo se asemeje mas a como funciona el algoritmo BL. Para soluciona esto, propondremos una bajada de temperatura uniforme realizado $T_k = T_k * 0.8$. Esto son los resultados:

```
Generados con temperatura K 12.4934: Exito = 62, Vecinos = 67
Generados con temperatura K 9.99472: Exito = 62, Vecinos = 63
Generados con temperatura K 7.99578: Exito = 62, Vecinos = 66
Generados con temperatura K 6.39662: Exito = 62, Vecinos = 66
Generados con temperatura K 5.1173: Exito = 62, Vecinos = 66
Generados con temperatura K 4.09384: Exito = 62, Vecinos = 72
Generados con temperatura K 3.27507: Exito = 62, Vecinos = 70
Generados con temperatura K 2.62006: Exito = 62, Vecinos = 77
Generados con temperatura K 2.09604: Exito = 62, Vecinos = 83
Generados con temperatura K 1.67684: Exito = 62, Vecinos = 86
Generados con temperatura K 1.34147: Exito = 62, Vecinos = 91
Generados con temperatura K 1.07317: Exito = 62, Vecinos = 95
Generados con temperatura K 0.85854: Exito = 62, Vecinos = 130
Generados con temperatura K 0.686832: Exito = 62, Vecinos = 158
Generados con temperatura K 0.549466: Exito = 62, Vecinos = 161
Generados con temperatura K 0.439572: Exito = 62, Vecinos = 216
Generados con temperatura K 0.351658: Exito = 62, Vecinos = 283
Generados con temperatura K 0.281326: Exito = 62, Vecinos = 438
Generados con temperatura K 0.225061: Exito = 50, Vecinos = 620
Generados con temperatura K 0.180049: Exito = 25, Vecinos = 620
Generados con temperatura K 0.144039: Exito = 18, Vecinos = 620
Generados con temperatura K 0.115231: Exito = 5, Vecinos = 620
Generados con temperatura K 0.092185: Exito = 1, Vecinos = 620
Generados con temperatura K 0.073748: Exito = 0, Vecinos = 620
Total Vecinos Generados = 6008 de 620
Numero de Evaluaciones = 6008 de 15000
Temperatura K obtenida = 0.0589984
```

```
Generados con temperatura K 13.8574: Exito = 34, Vecinos = 37
Generados con temperatura K 11.0859: Exito = 34, Vecinos = 36
Generados con temperatura K 8.86873: Exito = 34, Vecinos = 36
Generados con temperatura K 7.09499: Exito = 34, Vecinos = 37
Generados con temperatura K 5.67599: Exito = 34, Vecinos = 36
Generados con temperatura K 4.54079: Exito = 34, Vecinos = 35
Generados con temperatura K 3.63263: Exito = 34, Vecinos = 45
Generados con temperatura K 2.90611: Exito = 34, Vecinos = 43
Generados con temperatura K 2.32489: Exito = 34, Vecinos = 49
Generados con temperatura K 1.85991: Exito = 34, Vecinos = 52
Generados con temperatura K 1.48793: Exito = 34, Vecinos = 50
Generados con temperatura K 1.19034: Exito = 34, Vecinos = 48
Generados con temperatura K 0.952273: Exito = 34, Vecinos = 68
Generados con temperatura K 0.761818: Exito = 34, Vecinos = 75
Generados con temperatura K 0.609455: Exito = 34, Vecinos = 84
Generados con temperatura K 0.487564: Exito = 34, Vecinos = 90
Generados con temperatura K 0.390051: Exito = 34, Vecinos = 142
Generados con temperatura K 0.312041: Exito = 34, Vecinos = 247
Generados con temperatura K 0.249633: Exito = 32, Vecinos = 340
Generados con temperatura K 0.199706: Exito = 18, Vecinos = 340
Generados con temperatura K 0.159765: Exito = 10, Vecinos = 340
Generados con temperatura K 0.127812: Exito = 8, Vecinos = 340
Generados con temperatura K 0.10225: Exito = 1, Vecinos = 340
Generados con temperatura K 0.0817996: Exito = 1, Vecinos = 340
Generados con temperatura K 0.0654397: Exito = 0, Vecinos = 340
Total Vecinos Generados = 3590 de 340
Numero de Evaluaciones = 3590 de 15000
Temperatura K obtenida = 0.0523518
```

```
Generados con temperatura K 14.5258: Exito = 40, Vecinos = 41
Generados con temperatura K 11.6206: Exito = 40, Vecinos = 42
Generados con temperatura K 9.2965: Exito = 40, Vecinos = 42
Generados con temperatura K 7.4372: Exito = 40, Vecinos = 47
Generados con temperatura K 5.94976: Exito = 40, Vecinos = 45
Generados con temperatura K 4.75981: Exito = 40, Vecinos = 42
Generados con temperatura K 3.80784: Exito = 40, Vecinos = 47
Generados con temperatura K 3.04628: Exito = 40, Vecinos = 48
Generados con temperatura K 2.43702: Exito = 40, Vecinos = 56
Generados con temperatura K 1.94962: Exito = 40, Vecinos = 51
Generados con temperatura K 1.55969: Exito = 40, Vecinos = 57
Generados con temperatura K 1.24775: Exito = 40, Vecinos = 71
Generados con temperatura K 0.998204: Exito = 40, Vecinos = 78
Generados con temperatura K 0.798563: Exito = 40, Vecinos = 71
Generados con temperatura K 0.63885: Exito = 40, Vecinos = 104
Generados con temperatura K 0.51108: Exito = 40, Vecinos = 117
Generados con temperatura K 0.408864: Exito = 40, Vecinos = 163
Generados con temperatura K 0.327091: Exito = 40, Vecinos = 267
Generados con temperatura K 0.261673: Exito = 36, Vecinos = 400
Generados con temperatura K 0.209338: Exito = 30, Vecinos = 400
Generados con temperatura K 0.167471: Exito = 12, Vecinos = 400
Generados con temperatura K 0.133977: Exito = 8, Vecinos = 400
Generados con temperatura K 0.107181: Exito = 3, Vecinos = 400
Generados con temperatura K 0.085745: Exito = 0, Vecinos = 400
Total Vecinos Generados = 3789 de 400
Numero de Evaluaciones = 3789 de 15000
Temperatura K obtenida = 0.068596
```

Vemos que la temperatura baja de forma mas lenta hasta disminuir en 0 el numero de exitos obtenidos en una iteración. Los resultados de este nuevo SE0.8-mej modificado seran los siguientes:

ES0.8-mej	**Colposcopy**				**Ionosphere**				**Texture**			
	%clas	**%red**	**Agr.**	**T**	**%clas**	**%red**	**Agr.**	**T**	**%clas**	**%red**	**Agr.**	**T**
P1	69,86	88,70	85,73	48,6823	86,78	85,29	91,23	25,1290	87,72	85 88,40	75,9077	
P2	67,24	85,48	84,98	48,5082	85,40	88,23	93,40	23,6457	90,22	77,5	87,84	89,1259
P3	67,82	87,09	85,65	46,2856	85,40	91,17	92,73	25,1391	90	80	89,09	90,9842
P4	73,91	80,64	78,04	52,0776	88,25	82,35	89,74	26,2741	87,04	82,5	88,97	90,8072
P5	73,04	82,25	83,23	48,7621	82,91	88,23	92,68	23,0440	88,86	85	89,31	95,0844
Media	70,38	84,83	83,52	48,8631	85,75	87,05	91,96	24,6463	88,77	82	88,72	88,3818

Comparación	**Colposcopy**				**Ionosphere**				**Texture**			
	%clas	**%red**	**Agr.**	**T**	**%clas**	**%red**	**Agr.**	**T**	**%clas**	**%red**	**Agr.**	**T**
ES0.8-mej	70,38	84,83	83,52	48,8631	85,75	87,05	91,96	24,6463	88,77	82	88,72	88,3818
ES	72,73	80,64	80,38	34,4220	85,82	82,94	88,62	12,1947	88,36	82,50	90,38	50,49056

Vemos que las mejoras del nuevo SE0.8-mej son notables frente al anterior SE en casi todos los aspectos. El problema podría verse en el aumento de tiempo por parte de SE0.8-mej, ya que este debe de realizar mas iteraciones debido a la menor disminución de temperatura por iteración.

Otra propuesta de mejora del SE es el cambio de la disminución dependiendo del numero de éxitos obtenido en la iteración. Esto seria de la siguiente forma $T_k = T_k * \frac{exitos*0.8}{MaximoExitos}$. Esto mezclaría el anterior algoritmo, ya que la máxima disminución de temperatura seria de $T_k = T_k * 0.8$ en el caso de conseguir el numero máximo de éxitos y cuanto menos éxitos se obtengan mas rápida sera la disminución de temperatura, provocando que cuando $exitos = 0$, $T_k = 0$ también. Con esta disminución obtenemos tiempos menores a los dados en SE0.8-mej con unos resultados muy similares. Observemos como cambia la temperatura de este modo:

```
Generados con temperatura K 12.4934: Exito = 62, Vecinos = 67
Generados con temperatura K 9.99472: Exito = 62, Vecinos = 63
Generados con temperatura K 7.99578: Exito = 62, Vecinos = 66
Generados con temperatura K 6.39662: Exito = 62, Vecinos = 66
Generados con temperatura K 5.1173: Exito = 62, Vecinos = 66
Generados con temperatura K 4.09384: Exito = 62, Vecinos = 72
Generados con temperatura K 3.27507: Exito = 62, Vecinos = 70
Generados con temperatura K 2.62006: Exito = 62, Vecinos = 77
Generados con temperatura K 2.09604: Exito = 62, Vecinos = 83
Generados con temperatura K 1.67684: Exito = 62, Vecinos = 86
Generados con temperatura K 1.34147: Exito = 62, Vecinos = 91
Generados con temperatura K 1.07317: Exito = 62, Vecinos = 95
Generados con temperatura K 0.85854: Exito = 62, Vecinos = 130
Generados con temperatura K 0.686832: Exito = 62, Vecinos = 158
Generados con temperatura K 0.549466: Exito = 62, Vecinos = 161
Generados con temperatura K 0.439572: Exito = 62, Vecinos = 216
Generados con temperatura K 0.351658: Exito = 62, Vecinos = 283
Generados con temperatura K 0.281326: Exito = 62, Vecinos = 438
Generados con temperatura K 0.225061: Exito = 50, Vecinos = 620
Generados con temperatura K 0.145201: Exito = 14, Vecinos = 620
Generados con temperatura K 0.0262298: Exito = 0, Vecinos = 620
Total Vecinos Generados = 4148 de 620
Numero de Evaluaciones = 4148 de 15000
Temperatura K obtenida = 0
```

```
Generados con temperatura K 13.8574: Exito = 34, Vecinos = 37
Generados con temperatura K 11.0859: Exito = 34, Vecinos = 36
Generados con temperatura K 8.86873: Exito = 34, Vecinos = 36
Generados con temperatura K 7.09499: Exito = 34, Vecinos = 37
Generados con temperatura K 5.67599: Exito = 34, Vecinos = 36
Generados con temperatura K 4.54079: Exito = 34, Vecinos = 35
Generados con temperatura K 3.63263: Exito = 34, Vecinos = 45
Generados con temperatura K 2.90611: Exito = 34, Vecinos = 43
Generados con temperatura K 2.32489: Exito = 34, Vecinos = 49
Generados con temperatura K 1.85991: Exito = 34, Vecinos = 52
Generados con temperatura K 1.48793: Exito = 34, Vecinos = 50
Generados con temperatura K 1.19034: Exito = 34, Vecinos = 48
Generados con temperatura K 0.952273: Exito = 34, Vecinos = 68
Generados con temperatura K 0.761818: Exito = 34, Vecinos = 75
Generados con temperatura K 0.609455: Exito = 34, Vecinos = 84
Generados con temperatura K 0.487564: Exito = 34, Vecinos = 90
Generados con temperatura K 0.390051: Exito = 34, Vecinos = 142
Generados con temperatura K 0.312041: Exito = 34, Vecinos = 247
Generados con temperatura K 0.249633: Exito = 32, Vecinos = 340
Generados con temperatura K 0.187959: Exito = 16, Vecinos = 340
Generados con temperatura K 0.0707609: Exito = 0, Vecinos = 340
Total Vecinos Generados = 2230 de 340
Numero de Evaluaciones = 2230 de 15000
Temperatura K obtenida = 0
```

```
Generados con temperatura K 14.5258: Exito = 40, Vecinos = 41
Generados con temperatura K 11.6206: Exito = 40, Vecinos = 42
Generados con temperatura K 9.2965: Exito = 40, Vecinos = 42
Generados con temperatura K 7.4372: Exito = 40, Vecinos = 47
Generados con temperatura K 5.94976: Exito = 40, Vecinos = 45
Generados con temperatura K 4.75981: Exito = 40, Vecinos = 42
Generados con temperatura K 3.80784: Exito = 40, Vecinos = 47
Generados con temperatura K 3.04628: Exito = 40, Vecinos = 48
Generados con temperatura K 2.43702: Exito = 40, Vecinos = 56
Generados con temperatura K 1.94962: Exito = 40, Vecinos = 51
Generados con temperatura K 1.55969: Exito = 40, Vecinos = 57
Generados con temperatura K 1.24775: Exito = 40, Vecinos = 71
Generados con temperatura K 0.998204: Exito = 40, Vecinos = 78
Generados con temperatura K 0.798563: Exito = 40, Vecinos = 71
Generados con temperatura K 0.63885: Exito = 40, Vecinos = 104
Generados con temperatura K 0.51108: Exito = 40, Vecinos = 117
Generados con temperatura K 0.408864: Exito = 40, Vecinos = 163
Generados con temperatura K 0.327091: Exito = 40, Vecinos = 267
Generados con temperatura K 0.261673: Exito = 36, Vecinos = 400
Generados con temperatura K 0.188405: Exito = 26, Vecinos = 400
Generados con temperatura K 0.0979704: Exito = 1, Vecinos = 400
Generados con temperatura K 0.00195941: Exito = 1, Vecinos = 400
Total Vecinos Generados = 2989 de 400
Numero de Evaluaciones = 2989 de 15000
Temperatura K obtenida = 3.91882e-05
```

Visto esto, he decidido que dada la disminución del tiempo podríamos disminuir aun mas la bajada de temperatura, acción que también podríamos realizar en SE0.8-mej, pero estos tiempos resultan ser

algo mas altos sin una mejora tan significativa respecto a lo que se consigue realizando SE0.95-exito.

Como resultado obtenemos la siguiente tabla comparativa:

Comparación	Colposcopy				Ionosphere				Texture			
	%clas	%red	Agr.	T	%clas	%red	Agr.	T	%clas	%red	Agr.	T
ES0.95-exito	71,86	87,09	87,27	119,7612	85,11	91,17	92,74	52,8162	87,09	85	90,40	188,977
ES0.8-mej	70,38	84,83	83,52	48,86310	85,75	87,05	91,96	24,6463	88,77	82	88,72	88,3818
ES-0,8-exito	71,07	83,87	83,045	38,79922	85,54	86,47	91,52	18,1861	88,90	82,5	88,70	64,4367
ES	72,73	80,64	80,38	34,4220	85,82	82,94	88,62	12,1947	88,36	82,50	90,38	50,49056

El resultado es mejor que la modificación anterior pero esto supone un coste bastante alto en tiempo o una disminución algo menor de la calidad del algoritmo, en el caso de SE0.8-exito.

10 Experimento

Dado que hemos estudiado el uso de la búsqueda de entorno variable he decidido simularlo con el código usado para ILS. Las modificaciones serian las siguientes:

```
double calidadMutacion, mejorCalidad, tamEntorno = 1/numIteraciones;
int mutaciones = tamEntorno*col;
```

```
if(calidadMutacion > mejorCalidad){ //Si la calidad mejora, establecemos dicho w como el mejor
    mejorCalidad = calidadMutacion;
    w = wMutacion;
}else{
    tamEntorno += 1/numIteraciones;
    mutaciones = tamEntorno*col;
    cout << "Entorno Maximo Aumentado: " << tamEntorno << endl;
}
```

Esto proporciona que el numero de mutaciones para obtener vecinos aumente, proporcionando así vecinos mas distantes a la solución mejor y expandiendo mas el entorno de búsqueda. Los resultados son estos:

ILS / VSN	Colposcopy				Ionosphere				Texture			
	%clas	%red	Agr.	T	%clas	%red	Agr.	T	%clas	%red	Agr.	T
VSN	71,51	87,74	86,37	110,514	84,18	90,00	94,14	76,3602	88,04	86	91,09	254,482
ILS	70,99	91,29	90,42	120,783	85,39	90,58	93,30	76,5795	88,18	84,5	91,06	246,072

Observando la tabla vemos que la mejor en ínfima o empeora, en el caso de Colposcopy. Por tanto, dado que los tiempos son muy similares en algunos casos, concluimos que ILS se adapta mejor a problemas