Estimation of Distribution Algorithms Applied To Combinatorial Optimization Problems

Pedro Larrañaga (1), José A. Lozano (1), H. Mühlenbein (2)

(1) Departamento de Ciencias de la Computación e Inteligencia Artificial
Universidad del País Vasco - Euskal Herriko Unibertsitatea
Spain

(2) GMD Forschungszentrum Informationstechnik
Sank Agustin
Germany

e-mail: ccplamup@si.ehu.es, lozano@si.ehu.es, muehlenbein@gmd.de

Estimation of Distribution Algorithms (EDAs) are a new tool for Evolutionary Computation. Based on Genetic Algorithms (GAs) this new class of algorithms generalizes GAs by replacing the crossover and mutation operators by learning and sampling the probability distribution of the best individuals of the population at each iteration. In this paper we present an introduction to EDAs in the field of combinatorial optimization. The algorithms are organised taking the complexity of the probabilistic model used into account. We also provide some points to the literature.

Inteligencia Artificial, Revista Iberoamericana de Inteligencia Artificial. No.19 (2003),pp. 149-168
ISSN: 1137-3601.

Algoritmos de Estimación de Distribuciones en Problemas de Optimización Combinatoria

Pedro Larrañaga *
Departamento de Ciencias de la Computación e Inteligencia Artificial
Universidad del País Vasco-Euskal Herriko Unibertsitatea. Spain
ccplamup@si.ehu.es

Jose Antonio Lozano
Departamento de Ciencias de la Computación e Inteligencia Artificial
Universidad del País Vasco-Euskal Herriko Unibertsitatea. Spain
lozano@si.ehu.es

Heinz Mühlenbein
GMD Forschungszentrum Informationstechnik
Sank Agustin. Germany
muehlenbein@gmd.de

Resumen

Los algoritmos de estimación de distribuciones son un conjunto de métodos englobados dentro del paradigma de la computación evolutiva. Estos métodos se basan principalmente en sustituir el cruce y la mutación por la estimación y posterior muestreo de una distribución de probabilidad aprendida a partir de los individuos seleccionados. Este conjunto de algoritmos ha sido objeto de gran atención por parte de la comunidad científica alrededor de la computación evolutiva y los modelos gráficos probabilísticos. El presente artículo pretende introducir los algoritmos de estimación de distribuciones en el campo de la optimización combinatoria, al mismo tiempo que realizar una revisión exhaustiva de la bibliografía. La presentación de los diferentes algoritmos se realizará ordenándolos en base a la complejidad de las interrelaciones que son capaces de expresar los modelos probabilísticos que genera cada algoritmo.

1. Introducción

El comportamiento de los algoritmos de computación evolutiva más habituales (algoritmos genéticos y estrategias evolutivas) depende de varios parámetros asociados a los mismos (operadores de cruce y mutación, probabilidades de cruce y mutación, tamaño de la población, número de generaciones, tasa de reemplazamiento generacional, etc.). Si no se tiene experiencia en el uso del algoritmo evolutivo en el problema de optimización que se pretende resolver, la determinación de los valores adecuados para los parámetros anteriores se convierte en si mismo en un problema de optimización, tal y como fué expuesto por Grefenstette (1986). Este motivo, junto con el hecho de que la predicción de los movimientos

*Este trabajo ha sido subvencionado por el Ministerio de Ciencia y Tecnología, bajo la ayuda TIC2001-2973-C05-03

de la población de individuos en el espacio de búsqueda sea extremadamente difícil, ha motivado el nacimiento de un tipo de algoritmos conocidos como algoritmos de estimación de distribuciones (*Estimation of Distribution Algorithms (EDAs)*). Los EDAs fueron introducidos en el ámbito de la computación evolutiva por Mühlenbein y Paaß (1996). Aproximaciones similares previas pueden encontrarse en el libro de Zhigljavsky (1991).

Holland (1975) se percató de que el tener en consideración las variables que interactuaban podía ser beneficioso para los algoritmos genéticos. El explotar este tipo de información se conoce bajo el término de *linkage learning*. Estas mismas ideas han sido seguidas por otros autores, tales como Goldberg y col. (1989), Goldberg y col. (1993), Kargupta (1996), Kargupta y Goldberg (1997), Bandyopadyay y col. (1998), Lobo y col. (1998), van Kemenade (1998) y Bosman y Thierens (1999b). En los anteriores trabajos, los algoritmos genéticos se extienden para procesar los denominados *building blocks*, entendiendo por los mismos agrupaciones de variables que conviene que el algoritmo genético no destruya en su proceso evolutivo.

En contraposición con los algoritmos genéticos, los EDAs no requieren de operadores de cruce ni de mutación. La nueva población de individuos se obtiene por simulación de una distribución de probabilidad, la cual es estimada a partir de una base de datos conteniendo los individuos seleccionados en la generación anterior. Mientras que en el resto de algoritmos evolutivos las interrelaciones entre las variables (genes o posiciones de los individuos) se mantienen implícitas, en los EDAs dichas interrelaciones se expresan de manera explícita a través de la distribución de probabilidad conjunta asociada a los individuos seleccionados en cada generación. De hecho, dicha estimación de la distribución de probabilidad conjunta de los individuos seleccionados es (tal y como se verá a lo largo del trabajo) el verdadero *cuello de botella* de esta nueva aproximación.

El objetivo fundamental de este trabajo es el presentar una revisión de las distintas aproximaciones desarrolladas para los EDAs en el campo combinatorio. Para una revisión exhaustiva de los EDAs tanto en el campo combinatorio como en el continuo, el lector puede consultar los siguientes trabajos Larrañaga y col. (1999a, 1999b), y Pelikan y col. (1999b), así como el reciente libro de Larrañaga y Lozano (2002). Véase González y col. (2001) para una revisión de resultados teóricos relacionados con EDAs.

2. Un ejemplo sencillo

Con el objetivo de entender los distintos componentes y pasos de los EDAs, vamos a aplicar la versión más simple de los mismos a un ejemplo trivial de optimización.

Supongamos que estamos tratando de maximizar la función *OneMax* definida en un espacio de dimensión 6. Es decir, estamos tratando de obtener el máximo de la función: $h(\boldsymbol{x}) = \sum_{i=1}^{6} x_i$ con $x_i = 0, 1$.

La población inicial se obtiene al azar muestreando la siguiente distribución de probabilidad: $p_0(\boldsymbol{x}) = \prod_{i=1}^{6} p_0(x_i)$, donde $p_0(X_i = 1) = 0{,}5$ para $i = 1, \ldots, 6$. Esto significa que la distribución de probabilidad de la que estamos muestreando se factoriza como el producto de seis distribuciones de probabilidad marginales univariantes, cada una de las cuales sigue una distribución de Bernouilli con un parámetro $p = 0{,}5$. Denotamos por D_0 el fichero de 20 casos –véase Cuadro 1– obtenido a partir de esta simulación.

En un segundo paso, seleccionamos algunos individuos de D_0. Esto puede hacerse usando cualquiera de los métodos estándar de selección habituales en computación evolutiva. Supongamos que nuestro método de selección es truncación, y que seleccionamos la mitad de la población. Denotamos por D_0^{Se} el fichero de casos conteniendo a los individuos seleccionados. En caso de empates en la función de evaluación de varios individuos (por ejemplo los individuos numerados como 1, 9, 12, 18 y 20), la selección se efectúa de manera probabilística mediante una distribución uniforme. En este ejemplo, necesitamos escoger 3 individuos del conjunto de individuos cuya función de evaluación es 3.

Una vez que tenemos los 10 individuos seleccionados, D_0^{Se}, –véase Cuadro 2– se trata de expresar de manera explícita, usando la distribu-

Cuadro 1: La población inicial, D_0.

	X_1	X_2	X_3	X_4	X_5	X_6	$h(\boldsymbol{x})$
1	1	0	1	0	1	0	3
2	0	1	0	0	1	0	2
3	0	0	0	1	0	0	1
4	1	1	1	0	0	1	4
5	0	0	0	0	0	1	1
6	1	1	0	0	1	1	4
7	0	1	1	1	1	1	5
8	0	0	0	1	0	0	1
9	1	1	0	1	0	0	3
10	1	0	1	0	0	0	2
11	1	0	0	1	1	1	4
12	1	1	0	0	0	1	3
13	1	0	1	0	0	0	2
14	0	0	0	0	1	1	2
15	0	1	1	1	1	1	5
16	0	0	0	1	0	0	1
17	1	1	1	1	1	0	5
18	0	1	0	1	1	0	3
19	1	0	1	1	1	1	5
20	1	0	1	1	0	0	3

ción de probabilidad conjunta, las características de dichos individuos seleccionados. Aunque somos conscientes de que es una buena idea que dicha distribución de probabilidad conjunta tenga en cuenta todas las interdependencias entre las variables, en este ejemplo vamos a construir el modelo probabilístico más sencillo. En el mismo, cada variable se considera independiente del resto de las variables. Usando notación matemática, tenemos:

$$p_1(\boldsymbol{x}) = p_1(x_1, \ldots, x_6) = \prod_{i=1}^{6} p(x_i|D_0^{Se}).$$

Por tanto tan sólo necesitamos 6 parámetros para especificar el modelo. Cada parámetro, $p(x_i|D_0^{Se})$ con $i = 1, ..., 6$, se estimará a partir del fichero de casos D_0^{Se} por medio de sus frecuencias relativas correspondientes, $\hat{p}(X_i = 1|D_0^{Se})$.

Se obtienen los siguientes valores para los parámetros:

$$\hat{p}(X_1 = 1|D_0^{Se}) = 0{,}7 \quad \hat{p}(X_2 = 1|D_0^{Se}) = 0{,}7$$
$$\hat{p}(X_3 = 1|D_0^{Se}) = 0{,}6 \quad \hat{p}(X_4 = 1|D_0^{Se}) = 0{,}6$$
$$\hat{p}(X_5 = 1|D_0^{Se}) = 0{,}8 \quad \hat{p}(X_6 = 1|D_0^{Se}) = 0{,}7.$$

Muestreando esta distribución de probabilidad conjunta, $p_1(\boldsymbol{x})$, obtenemos una nueva

Cuadro 2: Los individuos seleccionados de la población inicial D_0^{Se}.

	X_1	X_2	X_3	X_4	X_5	X_6
1	1	0	1	0	1	0
4	1	1	1	0	0	1
6	1	1	0	0	1	1
7	0	1	1	1	1	1
11	1	0	0	1	1	1
12	1	1	0	0	0	1
15	0	1	1	1	1	1
17	1	1	1	1	1	0
18	0	1	0	1	1	0
19	1	0	1	1	1	1

Cuadro 3: La población de individuos en la primera generación, D_1.

	X_1	X_2	X_3	X_4	X_5	X_6	$h(\boldsymbol{x})$
1	1	1	1	1	1	1	6
2	1	0	1	0	1	1	4
3	1	1	1	1	1	0	5
4	0	1	0	1	1	1	4
5	1	1	1	1	0	1	5
6	1	0	0	1	1	1	4
7	0	1	0	1	1	0	3
8	1	1	1	0	1	0	4
9	1	1	1	0	0	1	4
10	1	0	0	1	1	1	4
11	1	1	0	0	1	1	4
12	1	0	1	1	1	0	4
13	0	1	1	0	1	1	4
14	0	1	1	1	1	0	4
15	0	1	1	1	1	1	5
16	0	1	1	0	1	1	4
17	1	1	1	1	1	0	5
18	0	1	0	0	1	0	2
19	0	0	1	1	0	1	3
20	1	1	0	1	1	1	5

Cuadro 4: Los individuos seleccionados, D_1^{Se}, de la población en su primera generación.

	X_1	X_2	X_3	X_4	X_5	X_6
1	1	1	1	1	1	1
2	1	0	1	0	1	1
3	1	1	1	1	1	0
5	1	1	1	1	0	1
6	1	0	0	1	1	1
8	1	1	1	0	1	0
9	1	1	1	0	0	1
15	0	1	1	1	1	1
17	1	1	1	1	1	0
20	1	1	0	1	1	1

población de individuos, D_1. En la Cuadro 3 se puede consultar los 20 individuos obtenidos, cada uno de los cuales es evaluado por medio de $h(\boldsymbol{x})$. De nuevo seleccionamos los 10 mejores individuos de D_1, obteniendo –tal y como puede verse en Cuadro 4– el fichero de casos D_1^{Se}.

A partir de este fichero de casos podemos obtener

$$p_2(\boldsymbol{x}) = p_2(x_1, \ldots, x_6) = \prod_{i=1}^{6} p(x_i|D_1^{Se})$$

donde $p(x_i|D_1^{Se})$ con $i = 1, ..., 6$ es estimada a partir de D_1^{Se} por medio de las frecuencias relativas correspondientes, $\hat{p}(X_i = 1|D_1^{Se})$.

En este caso los valores de los parámetros son:

$$\begin{array}{ll} \hat{p}(X_1 = 1|D_1^{Se}) = 0{,}9 & \hat{p}(X_2 = 1|D_1^{Se}) = 0{,}8 \\ \hat{p}(X_3 = 1|D_1^{Se}) = 0{,}8 & \hat{p}(X_4 = 1|D_1^{Se}) = 0{,}7 \\ \hat{p}(X_5 = 1|D_1^{Se}) = 0{,}8 & \hat{p}(X_6 = 1|D_1^{Se}) = 0{,}7. \end{array}$$

Estos tres pasos consistentes en (i) seleccionar algunos individuos de la población de individuos presente, (ii) estimar la distribución de probabilidad de los individuos seleccionados, y (iii) muestrear de la distribución de probabilidad estimada, se repiten hasta que se verifique un criterio de parada previamente establecido. En el pseudocódigo de la Figura 1, podemos consultar un esquema de la aproximación EDA. Al comienzo se generan M individuos al azar, por ejemplo a partir de una distribución uniforme para cada variable. Estos M individuos constituyen la población inicial, D_0, y cada uno de ellos es evaluado. En un primer paso, un número N ($N \leq M$) de individuos es seleccionado (normalmente aquellos con mejores

EDA
$D_0 \leftarrow$ Generar M individuos al azar

Repetir for $l = 1, 2, \ldots$ hasta
que se cumpla el criterio de parada

$D_{l-1}^{Se} \leftarrow$ Seleccionar $N \leq M$ individuos
de D_{l-1} acorde con el método de selección

$p_l(\boldsymbol{x}) = p(\boldsymbol{x}|D_{l-1}^{Se}) \leftarrow$ Estimar la
distribución de probabilidad
de los individuos seleccionados

$D_l \leftarrow$ Muestrear M individuos (la nueva
población) a partir de $p_l(\boldsymbol{x})$

Figura 1: Pseudocódigo de la aproximación EDA.

valores de función objetivo). A continuación, se efectúa la inducción del modelo probabilístico n–dimensional que mejor refleja las interdependencias entre las variables. En un tercer paso, se obtienen M nuevos individuos (la nueva población) a partir de la simulación de la distribución de probabilidad que se ha aprendido en el paso previo. Los tres pasos anteriores se repiten hasta que se verifique la condición de parada. Ejemplos de la misma son: evaluación de un número fijo de individuos o de poblaciones, uniformidad en la población generada, no mejora en relación con el mejor individuo obtenido en la generación previa, etc.

El problema principal con los EDAs radica en como estimar la distribución de probabilidad $p_l(\boldsymbol{x})$. Obviamente, la computación de todos los parámetros necesarios para especificar la distribución de probabilidad n–dimensional conjunta no es práctica, a nada que n crezca. La manera de abordar el problema es considerar que la distribución de probabilidad se factoriza de acuerdo a un modelo de probabilidad simplificado.

3. EDAs en optimización combinatoria

3.1. Introducción

En esta sección vamos a revisar distintas propuestas de EDAs para la optimización combinatoria. Hemos organizado la sección en base a la complejidad del modelo probabilístico usado para aprender las interdependencias entre las variables a partir de la base de datos conteniendo a los individuos seleccionados.

3.2. Sin dependencias

En todos los trabajos pertenecientes a esta categoría se asume que la distribución de probabilidad n–dimensional conjunta factoriza como un producto de n distribuciones de probabilidad univariantes e independientes. Es decir, $p_l(\boldsymbol{x}) = \prod_{i=1}^{n} p_l(x_i)$ –véase Figura 5 para una representación gráfica. Obviamente esta hipótesis de independencia está muy alejada de lo que ocurre en un problema difícil de optimización combinatoria, en el cual existen interdependencias entre las variables.

3.2.1. UMDA

Mühlenbein (1998) introdujo el *Univariate Marginal Distribution Algorithm* (UMDA). UMDA generaliza trabajos previos de Syswerda (1993), Eshelman y Schaffer (1993) y Mühlenbein y Voigt (1996).

El pseudocódigo para el UMDA puede consultarse en Figura 2. Como puede verse en la Figura 2 el modelo usado para estimar en cada generación la distribución de probabilidad conjunta a partir de los individuos seleccionados, $p_l(\boldsymbol{x})$, es lo más simple posible. De hecho la distribución de probabilidad conjunta se ha factorizado como producto de distribuciones univariantes independientes. Es decir:

$$p_l(\boldsymbol{x}) = p(\boldsymbol{x}|D_{l-1}^{Se}) = \prod_{i=1}^{n} p_l(x_i).$$

Cada distribución de probabilidad univariante

UMDA
$D_0 \leftarrow$ Generar M individuos (la población inicial) al azar

Repetir para $l = 1, 2, \ldots$ hasta que se verifique el criterio de parada

$D_{l-1}^{Se} \leftarrow$ Seleccionar $N \leq M$ individuos de D_{l-1} de acorde con un método de selección

$p_l(\boldsymbol{x}) = p(\boldsymbol{x}|D_{l-1}^{Se}) = \prod_{i=1}^{n} p_l(x_i) =$

$\prod_{i=1}^{n} \frac{\sum_{j=1}^{N} \delta_j(X_i = x_i|D_{l-1}^{Se})}{N} \leftarrow$
Estimar la distribución de probabilidad conjunta

$D_l \leftarrow$ Muestrear M individuos (la nueva población) a partir de $p_l(\boldsymbol{x})$

Figura 2: Pseudocódigo para UMDA.

se estima a partir de las frecuencias marginales:

$$p_l(x_i) = \frac{\sum_{j=1}^{N} \delta_j(X_i = x_i|D_{l-1}^{Se})}{N}$$

donde

$$\delta_j(X_i = x_i|D_{l-1}^{Se}) = \begin{cases} 1 & \text{si en el } j\text{-ésimo caso de } D_{l-1}^{Se},\ X_i = x_i \\ 0 & \text{en otro caso.} \end{cases}$$

Se pueden consultar Santana y Ochoa (1999) y Santana y col. (2000) para modificaciones en la fase de simulación del UMDA. En Inza y col. (2000) se puede ver una aplicación al problema de la selección de variables, mientras que Rivera (1999) aplica el UMDA para la búsqueda de un sistema clasificador. Análisis matemáticos del UMDA pueden consultarse en los trabajos de Mahnig y Mühlenbein (2000) y Mühlenbein y Mahnig (2000).

3.2.2. PBIL

El algoritmo PBIL (*Population Based Incremental Learning*) fué introducido por Baluja (1994), y posteriormente mejorado por Baluja y Caruana (1995), con el objetivo de obtener el

óptimo de una función definida en un espacio binario n–dimensional: $\Omega = \{0,1\}^n$. En cada generación, la población de individuos se representa como un vector de probabilidades:

$$p_l(\boldsymbol{x}) = (p_l(x_1), \ldots, p_l(x_i), \ldots, p_l(x_n))$$

donde $p_l(x_i)$ denota la probabilidad de obtener el valor 1 en la i-ésima componente de D_l, es decir en la población de individuos de la l-ésima generación.

El algoritmo funciona de la siguiente manera. En cada generación, usando el vector de probabilidades, $p_l(\boldsymbol{x})$, se obtienen M individuos. Cada uno de estos M individuos es evaluado y los N mejores ($N \leq M$) son seleccionados. Denotamos los mismos de la siguiente manera:

$$\boldsymbol{x}^l_{1:M}, \ldots, \boldsymbol{x}^l_{i:M}, \ldots, \boldsymbol{x}^l_{N:M}.$$

Estos individuos seleccionados son usados para actualizar el vector de probabilidades, usando una regla Hebbiana:

$$p_{l+1}(\boldsymbol{x}) = (1-\alpha)p_l(\boldsymbol{x}) + \alpha\frac{1}{N}\sum_{k=1}^{N}\boldsymbol{x}^l_{k:M}$$

donde $\alpha \in (0,1]$ es un parámetro del algoritmo. Nótese que PBIL tan sólo puede verse como una instancia de los EDAs en el caso en que $\alpha = 1$. En tal caso PBIL coincide con UMDA.

La Figura 3 muestra un pseudocódigo del algoritmo PBIL. El algoritmo PBIL ha recibido mucha atención por parte de la comunidad investigadora. Prueba de ello es el gran número de trabajos existentes en la literatura.

Baluja (1995) compara el algoritmo PBIL con otros seis heurísticos en diferentes problemas de optimización. También en Monmarché y col. (1999, 2000) pueden consultarse distintas comparaciones empíricas. Galić y Höhfeld (1996), Maxwell y Anderson (1999) y Gallagher (2000) aplican PBIL al problema de obtener los pesos óptimos en una red neuronal. Servais y col. (1997) y Kvasnicka y col. (1996) proponen una extensión de PBIL a espacios de búsqueda de cardinalidad genérica. Otros trabajos en los que se pueden ver modificaciones del método básico son Schmidt y col. (1999), Fyfe (1999) y Baluja (1997). Sukthankar y col. (1997) muestran una aplicación a un problema relacionado con vehículos inteligentes, Salustowicz y Schmidhuber (1997, 1998) muestran aplicaciones de PBIL

Obtener un vector inicial de probabilidades $p_0(\boldsymbol{x})$

while no convergencia **do**
begin

Usando $p_l(\boldsymbol{x})$ obtener M individuos: $\boldsymbol{x}^l_1, \ldots, \boldsymbol{x}^l_k, \ldots, \boldsymbol{x}^l_M$

Evaluar y ordenar $\boldsymbol{x}^l_1, \ldots, \boldsymbol{x}^l_k, \ldots, \boldsymbol{x}^l_M$

Seleccionar los N ($N \leq M$) mejores individuos: $\boldsymbol{x}^l_{1:M}, \ldots, \boldsymbol{x}^l_{k:M}, \ldots, \boldsymbol{x}^l_{N:M}$

Actualizar el vector de probabilidades $p_{l+1}(\boldsymbol{x}) = (p_{l+1}(x_1), \ldots, p_{l+1}(x_n))$

for $i = 1, \ldots n$ **do**
$p_{l+1}(x_i) =$
$(1-\alpha)p_l(x_i) + \alpha\frac{1}{N}\sum_{k=1}^{N} x^l_{i,k:M}$

end

Figura 3: Pseudocódigo del algoritmo PBIL.

a la programación genética e Inza y col. (2001c) presentan una aplicación en un dominio médico.

Trabajo teórico relacionado con el algoritmo PBIL incluye las siguientes referencias: Höhfeld y Rudolph (1997) demuestran la convergencia en media para funciones lineales, Berny (2000a) deriva el algoritmo desde un punto de vista de un sistema dinámico, González y col. (2001a) prueban la fuerte dependencia del algoritmo con respecto a los valores iniciales de los parámetros, y González y col. (2001b) analizan el algoritmo usando un sistema dinámico discreto. También puede consultarse la tesis doctoral de Juels (1997). Es reseñable asimismo que Thathachar y Sastry (1987) introducen a finales de los ochenta un algoritmo muy similar a PBIL.

3.2.3. cGA

Harik y col. (1998) presentan un algoritmo denominado *compact Genetic Algorithm* (cGA)

1. Inicializar el vector de probabilidades $p_0(\boldsymbol{x})$
   $p_0(\boldsymbol{x}) = p_0(x_1, \ldots, x_i, \ldots, x_n) =$
   $(p_0(x_1), \ldots, p_0(x_i), \ldots, p_0(x_n)) =$
   $(0{,}5, \ldots, 0{,}5, \ldots, 0{,}5)$

2. $l = l + 1$. Muestrear $p_l(\boldsymbol{x})$ con $l = 0, 1, \ldots$
   obtener dos individuos: $\overset{l}{1}, \boldsymbol{x}_2^l$

3. Evaluar y ordenar $\boldsymbol{x}_1^l$ y $\boldsymbol{x}_2^l$ obteniendo:
   $\boldsymbol{x}_{1:2}^l$ (el mejor de los dos) y
   $\boldsymbol{x}_{2:2}^l$ (el peor de los dos)

4. Actualizar el vector de probabilidades
   $p_l(\boldsymbol{x})$ hacia $\boldsymbol{x}_{1:2}^l$
   for $i = 1$ **to** n **do**
   if $x_{i,1:2}^l \neq x_{i,2:2}^l$ **then**
   if $x_{i,1:2}^l = 1$ **then** $p_l(x_i) = p_{l-1}(x_i) + \frac{1}{K}$
   if $x_{i,1:2}^l = 0$ **then** $p_l(x_i) = p_{l-1}(x_i) - \frac{1}{K}$

5. Comprobar si el vector de probabilidades
   $p_l(\boldsymbol{x})$ ha convergido
   for $i = 1$ **to** n **do**
   if $p_l(x_i) > 0$ y $p_l(x_i) < 1$ **then**
   volver al Paso 2

6. $p_l(\boldsymbol{x})$ representa la solución final

Figura 4: Pseudocódigo del algoritmo cGA.

que puede verse como un ejemplo de EDA univariado. El algoritmo (para una representación binaria) comienza inicializando un vector de probabilidades donde cada componente sigue una distribución de Bernouilli con parámetro $p = 0{,}5$. A continuación se generan dos individuos al azar a partir de este vector de probabilidades. Una vez evaluados los dos individuos, se efectúa una competición entre ambos. La competición se lleva a cabo a nivel de cada variable unidimensional, de tal manera que si para la i-ésima posición el individuo con mejor función de evaluación toma un valor diferente del que tiene el individuo con peor valor, entonces la i-ésima componente del vector de probabilidades incrementa o disminuye en una constante la i-ésima posición del vector de probabilidades en función de que la i-ésima componente del individuo vencedor sea respectivamente un uno o un cero. Este proceso de adaptación del vector de probabilidades hacia el individuo vencedor continúa hasta que el vector de probabilidades haya convergido. La Figura 4 muestra un pseudocódigo para el cGA.

La Figura 5 muestra una representación gráfica de los distintos modelos probabilísticos sin interdependencias.

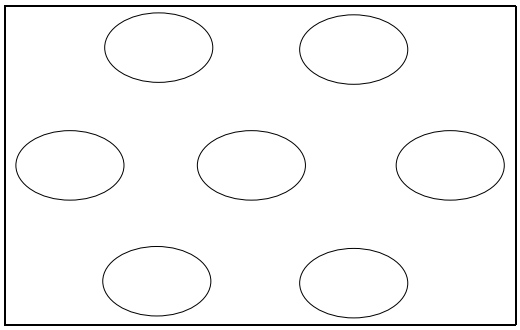

Figura 5: Representación gráfica de los modelos probabilísticos propuestos para los EDAs en optimización combinatoria sin considerar interdependencias entre las variables (UMDA, PBIL, cGA).

3.3. Dependencias bivariadas

La estimación de la distribución de probabilidad conjunta puede hacerse de manera rápida y sin necesidad de asumir independencia entre las variables, por medio de la consideración de dependencias entre pares de variables. En tal caso es suficiente con considerar estadísticos de orden dos. Mientras que en los algoritmos de la sección previa tan sólo se llevaba a cabo una estimación de los parámetros, ya que la estructura permanecía fija, en este apartado el aprendizaje paramétrico se extiende a estructural.

3.3.1. MIMIC

En De Bonet y col. (1997) se presenta un algoritmo voraz denominado MIMIC (*Mutual Information Maximization for Input Clustering*). MIMIC busca en cada generación la mejor permutación entre las variables con el objetivo de encontrar en cada generación la distribución de probabilidad $p_l^{\pi}(\boldsymbol{x})$, que se encuentra más cercana en divergencia de Kullback-Leibler a la

1. Seleccionar $i_n = \arg\min_j \hat{h}_l(X_j)$

2. **for** $k = n-1, \ldots, 1$

   Escoger $i_k = \arg\min_j \hat{h}_l(X_j \mid X_{i_{k+1}})$
   $j \neq i_{k+1}, \ldots, i_n$

3. $p_l^\pi(\boldsymbol{x}) =$
   $p_l(x_{i_1} \mid x_{i_2}) \cdots p_l(x_{i_{n-1}} \mid x_{i_n}) \cdot p_l(x_{i_n})$

Figura 6: La aproximación MIMIC para la estimación de la distribución de probabilidad en la generación l. Los símbolos $\hat{h}_l(X)$ y $\hat{h}_l(X \mid Y)$ denotan respectivamente la entropía empírica de X y la entropía empírica de X dado Y. Ambas son estimadas a partir de D_l^{Se}.

distribución empírica del conjunto de individuos seleccionados.

$$p_l^\pi(\boldsymbol{x}) = p_l(x_{i_1} \mid x_{i_2}) \cdots p_l(x_{i_{n-1}} \mid x_{i_n}) \cdot p_l(x_{i_n})$$

donde $\pi = (i_1, i_2, \ldots, i_n)$ denota una permutación de los índices $1, 2, \ldots, n$.

Se puede demostrar que la divergencia de Kullback-Leibler entre dos distribuciones de probabilidad, $p_l(\boldsymbol{x})$ y $p_l^\pi(\boldsymbol{x})$, se puede expresar como:

$$H_l^\pi(\boldsymbol{x}) = h_l(X_{i_n}) + \sum_{j=1}^{n-1} h_l(X_{i_j} \mid X_{i_{j+1}})$$

donde $h(X) = -\sum_x p(X = x) \log p(X = x)$ denota la entropía de Shannon de la variable X, y $h(X \mid Y) = \sum_y h(X \mid Y = y)p(Y = y)$, donde $h(X \mid Y = y) = -\sum_x p(X = x \mid Y = y) \log p(X = x|Y = y)$, denota la incertidumbre media en X dado Y. El problema de buscar la mejor $p_l^\pi(\boldsymbol{x})$ es equivalente a la búsqueda de la permutación π^* que minimiza $H_l^\pi(\boldsymbol{x})$. Para encontrar π^*, De Bonet y col. (1997) proponen un algoritmo voraz hacia adelante que evita el buscar en todo el espacio de permutaciones, cuya dimensión es $n!$. La idea es seleccionar X_{i_n} como la variable con menor entropía estimada, para a continuación en pasos sucesivos, seleccionar la variable –del conjunto de variables no seleccionadas hasta el momento– cuya entropía condicional media condicionada a la variable seleccionada en el paso anterior es mínima. La Figura 6 muestra el pseudocódigo para la estimación de la distribución de probabilidad conjunta efectuada por el algoritmo MIMIC.

3.3.2. COMIT

Baluja y Davies (1997a) propusieron un algoritmo que utiliza distribuciones de probabilidad basadas en relaciones bivariantes entre variables, para de esta forma generar un árbol de dependencia óptimo en términos de verosimilitud.

En este apartado nos vamos a concentrar en un algoritmo denominado COMIT (*Combining Optimizers with Mutual Information Trees*) el cual hibridiza la aproximación EDA con un optimizador local. COMIT se introdujo por Baluja y Davies (1997b, 1998). Puede consultarse su pseudocódigo en Figura 7. Nótese la diferencia entre esta aproximación y la presentada en la Figura 1 donde la estimación de la distribución de probabilidad conjunta es a través de los individuos seleccionados de la población usando el modelo estimado en la generación previa. La estimación de la distribución de probabilidad de los individuos seleccionados en cada generación se efectúa por medio de una red Bayesiana aprendida por medio del algoritmo propuesto por Chow y Liu (1968). Una vez que se ha aprendido el modelo probabilístico, la nueva generación de individuos se obtiene por muestreo del mismo. Los mejores de dichos individuos se consideran como puntos iniciales para llevar a cabo un procedimiento de búsqueda rápida. Algunos de los mejores individuos obtenidos como resultados de estas búsquedas rápidas se añaden a los individuos seleccionados en la generación previa para crear la nueva generación de individuos.

3.3.3. BMDA

Pelikan y Mühlenbein (1999) proponen una factorización de la distribución de probabilidad conjunta que tan sólo necesita de estadísticos de orden dos. Su aproximación, denominada BMDA (*Bivariate Marginal Distribution Algorithm*), está basada en la construcción de un grafo dirigido de dependencias acíclico, pero no necesariamente conectado. De hecho el grafo de dependencia puede verse como un conjunto de

COMIT
$D_0 \leftarrow$ Generar M individuos al azar

Repetir for $l = 1, 2, \ldots$ hasta que se verifique un criterio de parada

$D_{l-1}^{Se} \leftarrow$ Seleccionar $N \leq M$ individuos de D_{l-1} acorde con el método de selección

$p_l(\boldsymbol{x}) = p(\boldsymbol{x}|D_{l-1}^{Se}) = \prod_{i=1}^{n} p_l(x_i \mid x_{j(i)}) \leftarrow$ Estimar la distribución de probabilidad de los individuos seleccionados usando el algoritmo MWST de Chow and Liu (1968)

$D_l^{Sa} \leftarrow$ Muestrear M_1 individuos de $p_l(\boldsymbol{x})$

$D_l^{F-S} \leftarrow$ Obtener $M - N$ individuos por medio de un procedimiento rápido inicializado con la mejor solución de D_l^{Sa}

$D_l \leftarrow D_l^{F-S} \cup D_{l-1}^{Se}$

Figura 7: Pseudocódigo del algoritmo COMIT.

árboles que no están necesariamente conectados entre sí.

La idea básica subyacente a la construcción del grafo de dependencia es simple. En primer lugar se escoge una variable arbitraria y se añade la misma como un nodo del grafo. A continuación se introduce aquella variable que presente mayor dependencia –medida la misma a partir del estadístico χ^2 de Pearson– con respecto a la seleccionada en el paso previo. En el siguiente paso, necesitamos añadir al grafo la variable –que perteneciendo al conjunto de variables no seleccionadas hasta el momento– tenga mayor dependencia en relación a cualquiera de las variables previamente incorporadas al grafo. Este paso se repite hasta que todas las medidas de dependencias entre pares de variables –una de las cuales está en el grafo, mientras que la otra pertenece al conjunto de variables no seleccionadas– no superen un umbral previamente determinado. En tal momento, se debe seleccionar al azar una variable de entre el conjunto de variables que no pertenecen al grafo en ese momento. Todo el proceso se debe de repetir hasta que todas las variables se añadan al grafo.

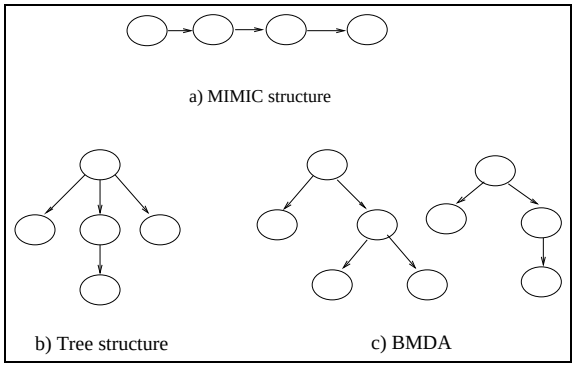

Figura 8: Representación gráfica de los modelos probabilísticos para EDAs con dependencias a pares (MIMIC, COMIT, BMDA).

En cada generación la factorización obtenida por el BMDA es de la forma siguiente:

$$p_l(\boldsymbol{x}) = \prod_{X_r \in R_l} p_l(x_r) \prod_{X_i \in V \setminus R_l} p_l(x_i \mid x_{j(i)})$$

donde V denota el conjunto de n variables, R_l denota el conjunto conteniendo la variable raíz –en la generación l– por cada una de las componentes conectadas del grafo de dependencia, y $X_{j(i)}$ representa la variable conectada a la variable X_i y añadida antes que X_i.

Las probabilidades de los nodos raíces, $p_l(x_r)$, al igual que las probabilidades condicionadas, $p_l(x_i \mid x_{j(i)})$, se estiman a partir de la base de datos, D_{l-1}^{Se}, conteniendo los individuos seleccionados.

La Figura 8 presenta una representación de los EDAs con dependencias a pares.

3.4. Dependencias múltiples

En la literatura se han propuesto varios algoritmos EDAs en los cuales la factorización de la distribución de probabilidad conjunta requiere de estadísticos de orden superior a dos.

El primer trabajo en el que se contempla la posibilidad de adaptar los métodos usados en la inducción de modelos gráficos probabilísticos a partir de datos en un algoritmo EDA fue presentando por Baluja y Davies (1997a). Sin embargo los autores no muestran ninguna evidencia de implementación de su propuesta.

La mayoría de estos algoritmos utilizan redes Bayesianas para codificar las distribución de probabilidad en cada paso. El lector interesado en los modelos gráficos probabilísticos puede consultar Castillo y col.(1999) para una amplia introducción a los mismos.

3.4.1. EcGA

Harik (1999) presenta un algoritmo –*Extended compact Genetic Algorithm* (EcGA)– cuya idea básica consiste en usar en cada generación un modelo de factorización de la distribución de probabilidad conjunta como producto de marginales de tamaño variable. Dicho modelo se induciría en cada generación del conjunto de individuos seleccionados. Las distribuciones de probabilidad marginales de tamaño variable se relacionan con las variables que están en el mismo grupo. El agrupamiento de variables se lleva a cabo por medio de un algoritmo voraz hacia adelante gracias al cual se obtiene una partición de las n variables. Cada grupo de variables se considera que es independiente del resto –tal y como se muestra en la Figura 11–. De esta manera la factorización de la distribución de probabilidad conjunta de las n variables es de la forma:

$$p_l(\boldsymbol{x}) = \prod_{c \in C_l} p_l(\boldsymbol{x}_c)$$

donde C_l denota el conjunto de grupos de variables en la l-ésima generación, y $p_l(\boldsymbol{x}_c)$ representa la distribución de probabilidad marginal de las variables $\boldsymbol{X}_c$, es decir de las variables que pertenecen al c-ésimo grupo en la l-ésima generación.

Como el algoritmo EcGA obtiene una partición del conjunto de las n variables, se tiene que para todo l y para todo $c, k \in C_l$:

$$\bigcup_{c \in C_l} \boldsymbol{X}_c = \{X_1, \ldots, X_n\}, \quad \boldsymbol{X}_c \cap \boldsymbol{X}_k = \emptyset.$$

El algoritmo voraz que busca los grupos de variables comienza con una partición con n grupos (una variable en cada grupo). A continuación el algoritmo lleva a cabo la unión de las dos variables que proporcionan la mayor reducción de una medida que conjuga la suma de las entropías de las distribuciones marginales con una penalización de la complejidad basada en el principio de descripción de longitud mínima (Rissanen, 1978).

La medida que EcGA trata de minimizar en cada generación tiene dos componentes:

- La *complejidad de la población comprimida* definida usando las entropías de las distribuciones marginales, como sigue:

$$N \sum_{c \in C_l} h(\boldsymbol{X}_c) =$$

$$-N \sum_{c \in C_l} \sum_{\boldsymbol{x}_c} p(\boldsymbol{X}_c = \boldsymbol{x}_c) \log p(\boldsymbol{X}_c = \boldsymbol{x}_c)$$

  y

- La *complejidad del modelo* la cual tiene en cuenta la dimensión del modelo de la manera siguiente:

$$\log N \sum_{c \in C_l} \dim \boldsymbol{X}_c$$

  donde $\dim \boldsymbol{X}_c$ representa el número de parámetros necesarios para especificar la distribución marginal $\boldsymbol{X}_c$. En el caso de que todas las variables unidimensionales pertenecientes al grupo c-ésimo fuesen binarias, se obtendría $\dim \boldsymbol{X}_c = 2^{|\boldsymbol{X}_c|} - 1$.

Teniendo en cuenta ambos componentes, la medida que EcGA trata de minimizar en cada generación es:

$$-N \sum_{c \in C_l} \sum_{\boldsymbol{x}_c} p(\boldsymbol{X}_c = \boldsymbol{x}_c) \log p(\boldsymbol{X}_c = \boldsymbol{x}_c) +$$

$$\log N \sum_{c \in C_l} \dim \boldsymbol{X}_c.$$

Esta medida fué denominada *complejidad combinada* por Harik (1999). El algoritmo de búsqueda voraz utilizado por EcGA comienza cada generación suponiendo que todas las variables son independientes. En cada paso el algoritmo trata de juntar los dos grupos cuya unión tiene un valor en complejidad combinada menor. Este proceso continua hasta que ninguna posible unión es capaz de proporcionar una reducción en la anterior medida. El modelo de producto marginal resultante es el que será usado en esa generación.

Tal y como puede verse en la Figura 9, la selección por torneo se utiliza en cada generación

EcGA
$D_0 \leftarrow$ Generar M individuos al azar

Repetir for $l = 1, 2, \ldots$ hasta que se verifique el criterio de parada

$D_{l-1}^{Se} \leftarrow$ Seleccionar $N \leq M$ individuos de D_{l-1} usando el método de selección por torneo

$p_l(\boldsymbol{x}) = p(\boldsymbol{x}|D_{l-1}^{Se}) = \prod_{c \in C_l} p_l(\boldsymbol{x}_c|D_{l-1}^{Se})$
Estimar la distribución de probabilidad de los individuos seleccionados por medio de un modelo de producto de marginales. El modelo se busca de manera voraz tratando de minimizar:
$-N \sum_{c \in C_l} \sum_{\boldsymbol{x}_c} p(\boldsymbol{X}_c = \boldsymbol{x}_c) \log p(\boldsymbol{X}_c = \boldsymbol{x}_c) +$
$\log N \sum_{c \in C_l} \dim \boldsymbol{X}_c$

$D_l \leftarrow$ Muestrear M individuos de $p_l(\boldsymbol{x})$

Figura 9: Pseudocódigo del algoritmo EcGA.

para obtener el conjunto de individuos seleccionados.

En Sastry y Goldberg (2000) se pueden consultar resultados experimentales relacionando el tamaño de la población con el tiempo de convergencia del EcGA.

3.4.2. Algoritmo FDA

En el trabajo de Mühlenbein y col.(1999) se introduce el algoritmo FDA (*Factorized Distribution Algorithm*). Este algoritmo se aplica a funciones aditivamente descomponibles para las cuales usando la *running intersection property* (Lauritzen,1996), se obtiene una factorización de la distribución de probabilidad conjunta basada en residuales $\boldsymbol{x}_{b_i}$, y separadores, $\boldsymbol{x}_{c_i}$.

Una función $h(\boldsymbol{x})$ es aditivamente descomponible si:

$$h(\boldsymbol{x}) = \sum_{s_i \in S} h_i(\boldsymbol{x}_{s_i})$$

donde el conjunto $S = \{s_1, \ldots s_k\}$, con $s_i \subset \{1, \ldots, n\}$, constituye un cubrimiento de $\{1, \ldots, n\}$, y los siguientes conjuntos:

$$d_i = \cup_{j=1}^{i} s_j$$
$$b_i = s_i \backslash d_{i-1}$$
$$c_i = s_i \cap d_{i-1}$$

satisfacen las tres condiciones siguientes:

- $b_i \neq \emptyset$ para todo $i = 1, \ldots, k$
- $d_k = \{1, 2, \ldots, n\}$
- $\forall\ i \geq 2\ \exists\ j < i$ tal que $c_i \subseteq s_j$.

En este caso la distribución de probabilidad conjunta puede factorizarse de la siguiente manera:

$$p_l(\boldsymbol{x}) = \prod_{i=1}^{k} p_l(\boldsymbol{x}_{b_i}|\boldsymbol{x}_{c_i}).$$

Esta factorización permanece válida para todas las iteraciones. Tan sólo se efectúan cambios en la estimación de las probabilidades las cuales en cada generación se efectúan de una base de datos conteniendo a los individuos seleccionados. En cualquier caso el requerimiento de la especificación de la factorización de la distribución de probabilidad conjunta es una desventaja en el momento de aplicar el algoritmo FDA a problemas de optimización genéricos. En tales casos además de un aprendizaje paramétrico es deseable un aprendizaje estructural.

Resultados teóricos para el FDA pueden consultarse en Mühlenbein y Mahnig (1999a, 1999b, 1999c, 2000), Zhang y Mühlenbein (1999) y Mahnig y Mühlenbein (2000).

3.4.3. PADA

En Soto y col. (1999) la factorización se lleva a cabo por medio de una red Bayesiana con estructura de poliárbol (no existe más de un camino no dirigido entre cada par de variables). El algoritmo propuesto se denomina PADA (*Polytree Approximation of Distribution Algorithms*) y puede ser considerado un híbrido entre los métodos basados en detectar (in)dependencias condicionales y aquellos que se engloban dentro de los métodos denominados score + búsqueda.

3.4.4. EBNA$_{PC}$, EBNA$_{K2+pen}$, EBNA$_{BIC}$

En el trabajo de Etxeberria y Larrañaga (1999) y Larrañaga y col. (2000a) la factorización de la distribución de probabilidad conjunta se codifica por medio de una red Bayesiana que se induce en cada generación a partir de la base de datos conteniendo los individuos seleccionados. El algoritmo se denomina EBNA (*Estimation of Bayesian Networks Algorithm*). Tal y como puede verse en la Figura 10, la red Bayesiana que corresponde a la primera iteración tiene como estructura un grafo sin arcos. En este caso la factorización de la distribución de probabilidad conjunta se obtiene a partir del producto de las n distribuciones uniformes de probabilidad marginales. Esto significa que la red Bayesiana inicial, BN_0, asigna la misma probabilidad a todos los puntos del espacio de búsqueda.

Teniendo en cuenta que necesitamos encontrar una estructura de modelo adecuada lo mas rápido posible, es deseable un algoritmo simple que devuelva una buena estructura –aunque esta no sea la óptima–. Un algoritmo con estas caracteríisticas es el algoritmo B (Buntine, 1991). El algoritmo B es un algoritmo voraz que comienza con una estructura sin arcos y en cada paso añade el arco que proporciona la mayor mejora en la medida de bondad que se esté utilizando. El algoritmo para cuando la adición de cualquier arco no incrementa la medida de bondad utilizada. Otra posibilidad para encontrar buenos modelos de manera rápida es el uso de estrategias de búsqueda local. En contraposición con el algoritmo B que comienza en cada paso desde el grafo sin arcos, las estrategias de búsqueda local comienzan la búsqueda con el modelo obtenido en la anterior generación. Se han implementado distintos criterios para guiar la búsqueda de buenas estructuras basados en diferentes scores –BIC, K2+pen– así como en el testeo de (in)dependencias condicionales entre tripletas de variables –algoritmo PC–, obteniendo de esta forma distintas instanciaciones de EBNA: EBNA$_{BIC}$, EBNA$_{K2+pen}$, EBNA$_{PC}$.

Nótese que el algoritmo EBNA$_{K2+pen}$, –Larrañaga y col. (2000a)– establece una cota superior al número de padres que la mejor estructura puede llegar a tener, para de esta forma controlar automáticamente la complejidad del modelo.

EBNA$_{PC}$, EBNA$_{K2+pen}$, EBNA$_{BIC}$

$BN_0 \leftarrow (S_0, \boldsymbol{\theta}^0)$ con S_0 DAG sin arcos y $\boldsymbol{\theta}^0$ uniforme. $p_0(\boldsymbol{x}) = \prod_{i=1}^{n} p(x_i) = \prod_{i=1}^{n} \frac{1}{r_i}$

$D_0 \leftarrow$ Muestrear M individuos de $p_0(\boldsymbol{x})$

Repetir for $l = 1, 2, \ldots$ hasta que se satisfaga el criterio de terminación

$D_{l-1}^{Se} \leftarrow$ Seleccionar N individuos de D_{l-1}

$S_l^* \leftarrow$ Encontrar la mejor estructura acorde con un criterio:
- tests de (in)dependencia condicional (EBNA$_{PC}$)
- Score Bayesiano penalizado + búsqueda (EBNA$_{K2+pen}$)
- Máxima verosimilitud penalizada + búsqueda (EBNA$_{BIC}$)

$\boldsymbol{\theta}^l \leftarrow$ Calcular θ_{ijk}^l usando D_{l-1}^{Se} como conjunto de datos

$BN_l \leftarrow (S_l^*, \boldsymbol{\theta}^l)$

$D_l \leftarrow$ Muestrear M individuos de BN_l using PLS

Figura 10: Pseudocódigo de los algoritmos EBNA$_{PC}$, EBNA$_{K2+pen}$, and EBNA$_{BIC}$.

Aplicaciones del algoritmo EBNA a diferentes problemas pueden consultarse en Bengoetxea y col. (2000, 2001a) –macheo inexacto de grafos–, Blanco y Lozano (2001) –optimización combinatoria–, de Campos y col. (2001) –inferencia abductiva parcial en redes Bayesianas–, Inza y col. (2000, 2001a, 2001c) –selección de subconjuntos de variables–, Inza y col. (2001b) –pesado de variables en K-NN–, Lozano y Mendiburu (2001) –planificación de trabajos–, Sierra y col. (2001) –inducción de reglas–, Robles y col. (2001) –problema del viajante de comercio– Roure y col. (2001) –agrupamiento particional– y Sagarna y Larrañaga (2001) –problema de la mochila–. Véase asimismo una versión paralela de EBNA en Sagarna (2000) y Lozano y col. (2001).

3.4.5. BOA

Pelikan y col. (1999a, 2000a, 2000b) y Pelikan y Goldberg (2000c) propusieron el algoritmo BOA (*Bayesian Optimization Algorithm*). BOA usa la métrica BDe (*Bayesian Dirichlet equivalence*) para evaluar la bondad de cada estructura. Esta métrica Bayesiana tiene la propiedad de que asigna el mismo valor a estructuras que reflejan las mismas (in)dependencias condicionales. La búsqueda de la mejor estructura es voraz y comienza en cada generación en el grafo sin arcos. Tratando de reducir la cardinalidad del espacio de búsqueda se asume la restricción de que cada nodo en la red Bayesiana tiene como mucho k padres. En Schwarz y Ocenasek (1999) se presentan comparaciones empíricas entre BOA y BMDA.

Véase Pelikan y Goldberg (2000b) para una modificación del algoritmo BOA para problemas jerárquicos usando un modelo híbrido denominado red de Huffman. En Pelikan y col. (2000c) BOA se adapta para incluir estructuras locales por medio de grafos de decisión, los cuales sirven para guiar la construcción de la red Bayesiana.

Otros trabajos que usan aproximaciones Bayesianas en la aproximación EDA –aunque en este caso el paradigma de red Bayesiana no se usa– son Zhang (1999), Zhang y Cho (2000) y Zhang y Shin (2000).

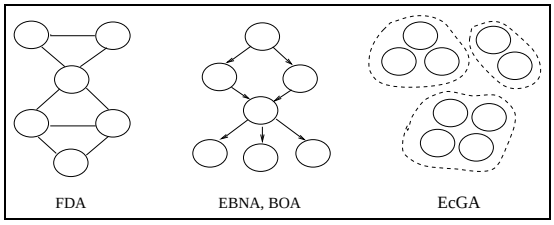

Figura 11: Representación gráfica de los modelos de probabilidad para los EDAs propuestos en optimización combinatoria con dependencias múltiples (FDA, EBNA, BOA y EcGA).

3.4.6. LFDA, FDA$_L$, FDA-BC, FDA-SC

Mühlenbein y Mahnig (1999c) introdujeron el algoritmo LFDA (*Learning Factorized Distribution Algorithm*) que básicamente sigue la misma aproximación que EBNA$_{BIC}$. La diferencia radica en que en LFDA la complejidad del modelo aprendido se controla por la métrica BIC en conjunción con una restricción acerca del máximo número de padres que cada variable puede llegar a tener en la red Bayesiana.

Ochoa y col. (1999) propone un algoritmo inicial FDA$_L$, para aprender –por medio de tests de (in)dependencia condicional– un árbol de unión a partir de una base de datos. La idea subyacente es obtener el árbol de unión que mejor satisface las (in)dependencias condicionales detectadas por los tests de (in)dependencia condicional.

En Ochoa y col. (2000a) se presenta un algoritmo de aprendizaje de estructura que tiene en cuenta cuestiones relativas a la fiabilidad y al coste computacional. El algoritmo se denomina FDA-BC (*Factorized Distribution Algorithm with Bayesian networks of Bounded Complexity*).

Ideas similares pueden consultarse en el algoritmo FDA-SC propuesto por Ochoa y col. (2000b). En este caso la factorización de la distribución de probabilidad conjunta se lleva a cabo por medio de estructuras simples (árboles, poliárboles, o bosques).

3.5. Modelos mixtos

Pelikan y Goldberg (2000a) proponen el uso de modelos probabilísticos más flexibles para estimar la distribución de probabilidad conjunta a partir de los individuos seleccionados. Para problemas simétricos y multimodales efectúan un agrupamiento de los individuos seleccionados en cada generación. Este agrupamiento se lleva a cabo en cada generación por medio de un método rápido (Forgy, 1965). El modelo obtenido puede escribirse como:

$$p_l(\boldsymbol{x}) = \sum_{i=1}^{k} \pi_{l,i} p_{l,i}(\boldsymbol{x})$$

donde en cada generación l, $\pi_{l,i}$ denota el peso de la i-ésima componente de la mixtura y $p_{l,i}(\boldsymbol{x})$ es la distribución de probabilidad del i-ésimo grupo obtenido a partir de los individuos seleccionados. Una de las restricciones del método –la cual puede ser resuelta por medio de la consideración de métodos de agrupamiento más sofisticados– es la determinación a priori del número de grupos. Tal y como comentan los autores, en problemas donde los picos de la función que se optimiza tienen tamaños desiguales, el método es muy sensible a métodos de selección que ejercen mucha presión.

En la misma línea, véase también el trabajo de Peña y col. (2001) en el cual una aproximación EDA basada en modelos mixtos, donde cada mixtura es una red Bayesiana y su correspondiente peso se obtiene a partir del algoritmo EM.

4. A modo de conclusión

En este trabajo se ha presentado una revisión de distintas aproximaciones EDAs en problemas de optimización combinatoria. Las distintas propuestas se han presentado usando una notación unificada y han sido organizadas desde el punto de vista de la complejidad del modelo gráfico probabilístico que aprenden en cada generación a partir de los datos.

El artículo ha pretendido al mismo tiempo poner al lector en disposición de comenzar a realizar trabajos de investigación sobre EDAs. Al ser éstos un paradigma de optimización relativamente nuevo, existen muchas campos abiertos donde llevar a cabo esta tarea de investigación. Dentro de estos campos podríamos destacar: modelado matemático de EDAs, optimización multiobjetivo utilizando EDAs, algoritmos EDAs paralelos, EDAs en funciones multimodales, nuevos campos de aplicación de EDAs, utilización de diferentes métodos a la hora de codificar la distribución de probabilidad, etc.

Referencias

[1] E. Alba, R. Santana, A. Ochoa, and M. Lazo. Finding typical testors by using an evolutionary strategy. In *Proceedings of the Fifth Ibero American Symposium on Pattern Recognition*, pages 267–278, 2000.

[2] S. Baluja. Population-based incremental learning: A method for integrating genetic search based function optimization and competitive learning. Technical Report CMU-CS-94-163, Carnegie Mellon University, 1994.

[3] S. Baluja. An empirical comparison of seven iterative and evolutionary function optimization heuristics. Technical Report CMU-CS-95-193, Carnegie Mellon University, 1995.

[4] S. Baluja. Genetic algorithms and explicit search statistics. *Advances in Neural Information Processing Systems*, 9:319–325, 1997.

[5] S. Baluja and R. Caruana. Removing the genetics from standard genetic algorithm. In A. Prieditis and S. Russell, editors, *Proceedings of the International Conference on Machine Learning*, pages 38–46. Morgan Kaufmann, 1995.

[6] S. Baluja and S. Davies. Combining multiple optimization runs with optimal dependency trees. Technical Report CMU-CS-97-157, Carnegie Mellon University, 1997.

[7] S. Baluja and S. Davies. Using optimal dependency-trees for combinatorial optimization: Learning the structure of the

search space. Technical Report CMU-CS-97-107, Carnegie Mellon University, 1997.

[8] S. Baluja and S. Davies. Fast probabilistic modeling for combinatorial optimization. In *AAAI-98*, 1998.

[9] S. Bandyopadhyay, H. Kargupta, and G. Wang. Revisiting the gemga: Scalable evolutionary optimization through linkage learning. In *Proceedings of the 1998 IEEE International Conference on Evolutionary Computation*, pages 603–608. IEEE Press, 1998.

[10] E. Bengoetxea, P. Larrañaga, I. Bloch, and A. Perchant. Solving graph matching with EDAs using a permutation–based representation. In P. Larrañaga and J. A. Lozano, editors, *Estimation of Distribution Algorithms. A New Tool for Evolutionary Computation*. Kluwer Academic Publishers, 2001.

[11] E. Bengoetxea, P. Larrañaga, I. Bloch, A. Perchant, and C. Boeres. Inexact graph matching using learning and simulation of Bayesian networks. An empirical comparison between different approaches with synthetic data. In *Workshop Notes of CaNew2000: Workshop on Bayesian and Causal Networks: From Inference to Data Mining*, 2000. Fourteenth European Conference on Artificial Intelligence, ECAI2000. Berlin.

[12] A. Berny. Selection and reinforcement learning for combinatorial optimization. In M. Schoenauer, K. Deb, G. Rudolph, X. Yao, E. Lutton, J. J. Merelo, and H.-P. Schwefel, editors, *Parallel Problem Solving from Nature – PPSN VI. Lecture Notes in Computer Science 1917*, pages 601–610, 2000.

[13] R. Blanco and J. A. Lozano. Empirical comparison of Estimation of Distribution Algorithms in combinatorial optimization. In P. Larrañaga and J. A. Lozano, editors, *Estimation of Distribution Algorithms. A New Tool for Evolutionary Computation*. Kluwer Academic Publishers, 2001.

[14] P. A.Ñ. Bosman and D. Thierens. Linkage information processing in distribution estimation algorithms. In W. Banzhaf, J. Daida, A. E. Eiben, M. H. Garzon, V. Honavar, M. Jakiela, and R. E. Smith, editors, *Proceedings of the Genetic and Evolutionary Computation Conference GECCO-99*, volume 1, pages 60–67. Morgan Kaufmann Publishers, 1999. San Francisco, LA.

[15] W. Buntine. Theory refinement in Bayesian networks. In *Proceedings of the Seventh Conference on Uncertainty in Artificial Intelligence*, pages 52–60, 1991.

[16] E. Castillo, J. M. Gutiérrez, and A. S. Hadi. *Expert Systems and Probabilistic Network Models*. Springer-Verlag, New York, 1997.

[17] C. Chow and C. Liu. Approximating discrete probability distributions with dependence trees. *IEEE Transactions on Information Theory*, 14:462–467, 1968.

[18] J. S. De Bonet, C. L. Isbell, and P. Viola. MIMIC: Finding optima by estimating probability densities. *Advances in Neural Information Processing Systems*, Vol. 9, 1997.

[19] L. M. de Campos, J. A. Gámez, P. Larrañaga, S. Moral, and T. Romero. Partial abductive inference in Bayesian networks: an empirical comparison between GAs and EDAs. In P. Larrañaga and J. A. Lozano, editors, *Estimation of Distribution Algorithms. A New Tool for Evolutionary Computation*. Kluwer Academic Publishers, 2001.

[20] L. J. Eshelman and J. D. Schaffer. Productive recombination and propagating and preserving schemata. *Foundations of Genetic Algorithms*, 3:299–314, 1993.

[21] R. Etxeberria and P. Larrañaga. Global optimization with Bayesian networks. In *II Symposium on Artificial Intelligence. CIMAF99. Special Session on Distributions and Evolutionary Optimization*, pages 332–339, 1999.

[22] E. Forgy. Cluster analysis of multivariate data: Efficiency versus interpretability of classifications. *Biometrics*, 21:768, 1965.

[23] C. Fyfe. Structured population-based incremental learning. *Soft Computing*, 2(4):191–198, 1999.

[24] E. Galić and M. Höhfeld. Improving the generalization performance of multi-layer-perceptrons with population-based incremental learning. In *Parallel Problem Solving from Nature. PPSN-IV*, pages 740–750, 1996.

[25] M. R. Gallagher. Multi-layer perceptron error surfaces: Visualization, structure and modelling. Technical Report Doctoral Thesis, Department of Computer Science and Electrical Engineering, University of Queensland, 2000.

[26] D. E. Goldberg, K. Deb, H. Kargupta, and G. Harik. Rapid, accurate optimization of difficult problems using fast messy genetic algorithms. In S. Forrest, editor, *Proceedings of the Fifth International Conference on Genetic Algorithms*, pages 56–64. Morgan Kauffman, 1993.

[27] D. E. Goldberg, B. Korb, and K. Deb. Messy genetic algorithms: Motivation, analysis and first results. *Complex Systems*, 3(5):493–530, 1989.

[28] C. González, J. A. Lozano, and P. Larrañaga. Analyzing the PBIL algorithm by means of discrete dynamical systems. *Complex Systems*, In press, 2001.

[29] C. González, J. A. Lozano, and P. Larrañaga. The converge behavior of PBIL algorithm: a preliminar approach. In V. Kŭrková, N. C. Steel, R.Ñeruda, and M. Kárný, editors, *International Conference on Artificial Neural Networks and Genetic Algorithms. ICANNGA-2001*, pages 228–231. Springer, 2001.

[30] C. González, J. A. Lozano, and P. Larrañaga. Mathematical modeling of Estimation of Distribution Algorithms. In P. Larrañaga and J. A. Lozano, editors, *Estimation of Distribution Algorithms. A New Tool for Evolutionary Computation*. Kluwer Academic Publishers, 2001.

[31] J. J. Grefenstette. Optimization of control parameters for genetic algorithms. *IEEE Transactions on Systems, Man, and Cybernetics*, 16(1):122–128, 1986.

[32] G. Harik. Linkage learning in via probabilistic modeling in the ECGA. Technical Report 99010, IlliGAL Technical Report, 1999.

[33] G. Harik, F. G. Lobo, and D. E. Golberg. The compact genetic algorithm. In *Proceedings of the IEEE Conference on Evolutionary Computation*, pages 523–528, 1998.

[34] M. Höhfeld and G. Rudolph. Towards a theory of population-based incremental learning. In *Proceedings of the 4th International Conference on Evolutionary Computation*, pages 1–5. IEEE Press, 1997.

[35] J. H. Holland. *Adaptation in Natural and Artificial Systems*. The University of Michigan Press, 1975.

[36] I. Inza, P. Larrañaga, and B. Sierra R. Etxeberria. Feature subset selection by Bayesian networks based optimization. *Artificial Intelligence*, 123(1–2):157–184, 2000.

[37] I. Inza, P. Larrañaga, and B. Sierra. Feature subset selection by Estimation of Distribution Algorithms. In P. Larrañaga and J. A. Lozano, editors, *Estimation of Distribution Algorithms. A New Tool for Evolutionary Computation*. Kluwer Academic Publishers, 2001.

[38] I. Inza, P. Larrañaga, and B. Sierra. Feature weighting in K-NN by means of Estimation of Distribution Algorithms. In P. Larrañaga and J. A. Lozano, editors, *Estimation of Distribution Algorithms. A New Tool for Evolutionary Computation*. Kluwer Academic Publishers, 2001.

[39] I. Inza, M. Merino, P. Larrañaga, J. Quiroga, B. Sierra, and M. Girala. Feature subset selection by population-based incremental learning. A case study in the survival of cirrhotic patients with TIPS. *Artificial Intelligence in Medicine*, In press, 2001.

[40] A. Juels. Topics in black-box combinatorial optimization. Technical Report Doctoral Thesis, University of California–Berkeley, 1997.

[41] H. Kargupta. The gene expression messy genetic algorithm. In *Proceedings of the 1996 IEEE International Conference on Evolutionary Computation*, pages 631–636. IEEE Press, 1996.

[42] H. Kargupta and D. E. Goldberg. Search, blackbox optimization, and sample complexity. In R. W. Belew and M. Vose, editors, *Foundations of Genetic Algorithms 4*. Morgan Kaufmann, 1997. San Mateo, CA.

[43] V. Kvasnicka, M. Pelikan, and J. Pospichal. Hill climbing with learning (an abstraction of genetic algorithms). *Neural Network World*, 6:773–796, 1996.

[44] P. Larrañaga, R. Etxeberria, J. A. Lozano, and J. M. Peña. Optimization by learning and simulation of Bayesian and Gaussian networks. Technical Report KZZA-IK-4-99, Department of Computer Science and Artificial Intelligence, University of the Basque Country, 1999.

[45] P. Larrañaga, R. Etxeberria, J. A. Lozano, and J. M. Peña. Combinatorial optimization by learning and simulation of Bayesian networks. In *Proceedings of the Sixteenth Conference on Uncertainty in Artificial Intelligence*, pages 343–352, 2000. Stanford.

[46] P. Larrañaga, R. Etxeberria, J. A. Lozano, B. Sierra, I. Inza, and J. M. Peña. A review of the cooperation between evolutionary computation and probabilistic graphical models. In *Second Symposium on Artificial Intelligence. Adaptive Systems. CIMAF 99*, pages 314–324, 1999. La Habana.

[47] P. Larrañaga and J.A. Lozano. *Estimation of Distribution Algorithms. A New Tool for Evolutionary Computation*. Kluwer Academic Publishers, 2001.

[48] S. L. Lauritzen. *Graphical Models*. Oxford University Press, 1996.

[49] F. G. Lobo, K. Deb, D. E. Goldberg, G. R. Harik, and L. Wang. Compressed introns in a linkage learning genetic algorithm. In *Genetic Programming 1998: Proceedings of the Third Annual Conference*, pages 551–558. Morgan Kauffman, 1998.

[50] J. A. Lozano, R. Sagarna, and P. Larrañaga. Parallel Estimation of Distribution Algorithms. In P. Larrañaga and J. A. Lozano, editors, *Estimation of Distribution Algorithms. A New Tool for Evolutionary Computation*. Kluwer Academis Publishers, 2001.

[51] T. Mahnig and H. Mühlenbein. Mathematical analysis of optimization methods using search distributions. In A. S. Wu, editor, *Proceedings of the 2000 Genetic and Evolutionary Computation Conference Workshop Program*, pages 205–208, 2000.

[52] B. Maxwell and S. Anderson. Training hidden Markov models using population-based learning. In *Genetic and Evolutionary Computation Conference, GECCO-99*, 1999.

[53] A. Mendiburu and J. A. Lozano. Solving job schedulling with Estimation of Distribution Algorithms. In P. Larrañaga and J. A. Lozano, editors, *Estimation of Distribution Algorithms. A New Tool for Evolutionary Computation*. Kluwer Academic Publishers, 2001.

[54] N. Monmarché, E. Ramat, L. Desbarats, and G. Venturini. Probabilistic search with genetic algorithms and ant colonies. In A. S. Wu, editor, *Proceedings of the 2000 Genetic and Evolutionary Computation Conference Workshop Program*, pages 209–211, 2000.

[55] N. Monmarché, E. Ramat, G. Dromel, M. Slimane, and G. Venturini. On the similarities between AS, BSC and PBIL: toward the birth of a new meta–heuristics. Technical Report 215, E3i, Université de Tours, 1999.

[56] H. Mühlenbein. The equation for response to selection and its use for prediction. *Evolutionary Computation*, 5:303–346, 1998.

[57] H. Mühlenbein and T. Mahnig. Convergence theory and applications of the factorized distribution algorithm. *Journal of Computing and Information Technology*, 7:19–32, 1999.

[58] H. Mühlenbein and T. Mahnig. The Factorized Distribution Algorithm for additively decomposed functions. In *Second Symposium on Artificial Intelligence. Adaptive Systems. CIMAF 99*, pages 301–313, 1999. La Habana.

[59] H. Mühlenbein and T. Mahnig. FDA - a scalable evolutionary algorithm for the optimization of additively decomposed functions. *Evolutionary Computation*, 7(4):353–376, 1999.

[60] H. Mühlenbein and T. Mahnig. Evolutionary algorithms: From recombination to search distributions. *Theoretical Aspects of Evolutionary Computing. Natural Computing*, pages 137–176, 2000.

[61] H. Mühlenbein, T. Mahnig, and A. Ochoa. Schemata, distributions and graphical models in evolutionary optimization. *Journal of Heuristics*, 5:215–247, 1999.

[62] H. Mühlenbein and G. Paaß. From recombination of genes to the estimation of distributions I. Binary parameters. In *Lecture Notes in Computer Science 1411: Parallel Problem Solving from Nature - PPSN IV*, pages 178–187, 1996.

[63] H. Mühlenbein and H.-M. Voigt. Gene pool recombination in genetic algorithms. *Metaheuristics: Theory and applications*, pages 53–62, 1996.

[64] A. Ochoa, H. Mühlenbein, and M. Soto. Factorized Distribution Algorithm using Bayesian networks. In A. S. Wu, editor, *Proceedings of the 2000 Genetic and Evolutionary Computation Conference Workshop Program*, pages 212–215, 2000.

[65] A. Ochoa, H. Mühlenbein, and M. Soto. A Factorized Distribution Algorithm using single connected Bayesian networks. In M. Schoenauer, K. Deb, G. Rudolph, X. Yao, E. Lutton, J. J. Merelo, and H.-P. Schwefel, editors, *Parallel Problem Solving from Nature – PPSN VI. Lecture Notes in Computer Science 1917*, pages 787–796, 2000.

[66] A. Ochoa, M. Soto, R. Santana, J. Madera, and N. Jorge. The Factorized Distribution Algorithm and the junction tree: A learning perspective. In *Second Symposium on Artificial Intelligence. Adaptive Systems. CIMAF 99*, pages 368–377, 1999. La Habana.

[67] M. Pelikan and D. E. Goldberg. Genetic algorithms, clustering, and the breaking of symmetry. In M. Schoenauer, K. Deb, G. Rudolph, X. Yao, E. Lutton, J. J. Merelo, and H.-P. Schwefel, editors, *Parallel Problem Solving from Nature – PPSN VI. Lecture Notes in Computer Science 1917*, pages 385–394, 2000.

[68] M. Pelikan and D. E. Goldberg. Hierarchical problem solving and the Bayesian optimization algorithm. In D. Whitley, D. Goldberg, E. Cantú-Paz, L. Spector, I. Parmee, and H.-G. Beyer, editors, *Proceedings of the Genetic and Evolutionary Computation Conference*, pages 267–274. Morgan Kaufmann, 2000.

[69] M. Pelikan and D. E. Goldberg. Research on the Bayesian optimization algorithm. In A. S. Wu, editor, *Proceedings of the 2000 Genetic and Evolutionary Computation Conference Workshop Program*, pages 212–215, 2000.

[70] M. Pelikan, D. E. Goldberg, and E. Cantú-Paz. BOA: The Bayesian optimization algorithm. In W. Banzhaf, J. Daida, A. E. Eiben, M. H. Garzon, V. Honavar, M. Jakiela, and R. E. Smith, editors, *Proceedings of the Genetic and Evolutionary Computation Conference GECCO-99*, volume 1, pages 525–532. Morgan Kaufmann Publishers, San Francisco, CA, 1999. Orlando, FL.

[71] M. Pelikan, D. E. Goldberg, and E. Cantú-Paz. Bayesian optimization algorithm, population sizing, and time to convergence. In D. Whitley, D. Goldberg, E. Cantú-Paz, L. Spector, I. Parmee, and H.-G. Beyer, editors, *Proceedings of the Genetic and Evolutionary Computation Conference*, pages 275–282. Morgan Kaufmann, 2000.

[72] M. Pelikan, D. E. Goldberg, and E. Cantú-Paz. Linkage problem, distribution estimation and Bayesian networks. *Evolutionary Computation*, 8(3):311–340, 2000.

[73] M. Pelikan, D. E. Goldberg, and F. Lobo. A survey of optimization by building and using probabilistic models. Technical Report IlliGAL Report 99018, University of Illinois at Urbana-Champaing, 1999.

[74] M. Pelikan, D. E. Goldberg, and K. Sastry. Bayesian optimization algorithm, decision graphs, and Occam's razor. Technical Report IlliGAL Report 200020, University of Illinois at Urbana-Champaing, 2000.

[75] M. Pelikan and H. Mühlenbein. The bivariate marginal distribution algorithm. *Advances in Soft Computing-Engineering Design and Manufacturing*, pages 521–535, 1999.

[76] J. M. Peña, J. A. Lozano, and P. Larrañaga. Benefits of data clustering in multimodal function optimization via EDAs. In P. Larrañaga and J. A. Lozano, editors, *Estimation of Distribution Algorithms. A New Tool for Evolutionary Computation*, 2001.

[77] J. Rissanen. Modeling by shortest data description. *Automatica*, pages 465–471, 1978.

[78] J. Rivera. Using Estimation of Distribution Algorithms as an evolutive component of the XCS classifier system. Technical report, University of La Habana (In spanish), 1999.

[79] V. Robles, P. de Miguel, and P. Larrañaga. Solving the travelling salesman problem with Estimation of Distribution Algorithms. In P. Larrañaga and J. A. Lozano, editors, *Estimation of Distribution Algorithms. A New Tool for Evolutionary Computation*. Kluwer Academic Publishers, 2001.

[80] J. Roure, R. Sangüesa, and P. Larrañaga. Partitional clustering by means of Estimation of Distribution Algorithms. In P. Larrañaga and J. A. Lozano, editors, *Estimation of Distribution Algorithms. A New Tool for Evolutionary Computation*. Kluwer Academic Publishers, 2001.

[81] R. Sagarna. Parallelization of Estimation of Distribution Algorithms. Technical Report Master Thesis, University of the Basque Country, Department of Computer Science and Artificial Intelligence (In spanish), 2000.

[82] R. Sagarna and P. Larrañaga. Solving the knapsack problem with Estimation of Distribution Algorithms. In P. Larrañaga and J. A. Lozano, editors, *Estimation of Distribution Algorithms. A New Tool for Evolutionary Computation*. Kluwer Academis Publishers, 2001.

[83] R. Salustowicz and J. Schmidhuber. Probabilistic incremental program evolution. *Evolutionary Computation*, 5(2):123–141, 1997.

[84] R. Salustowicz and J. Schmidhuber. Learning to predict trough probabilistic incremental program evolution and automatic task decomposition. Technical Report Technical Report IDSIA–11–98, University of Lugano, 1998.

[85] R. Santana and A. Ochoa. Dealing with constraints with Estimation of Distribution Algorithms: The univariate case. In *Second Symposium on Artificial Intelligence. Adaptive Systems. CIMAF 99*, pages 378–384, 1999. La Habana.

[86] R. Santana, A. Ochoa, M. Soto, F. B. Pereira, P. Machado, E. Costa, and A. Cardoso. Probabilistic evolution and the busy beaver problem. In D. Whitley, D. Goldberg, E. Cantú-Paz, L. Spector, I. Parmee, and H.-G. Beyer, editors, *Proceedings of the Genetic and Evolutionary Computation Conference*, pages 380–380. Morgan Kaufmann, 2000.

[87] K. Sastry and D. E. Goldberg. On extended compact genetic algorithm. In *GECCO-2000, Late Breaking Papers, Genetic and evolutionary Computation Conference*, pages 352–359, 2000.

[88] M. Schmidt, K. Kristensen, and T.R. Jensen. Adding genetics to the standar PBIL algorithm. In *Congress on Evolutionary Computation. CEC'99*, 1999.

[89] J. Schwarz and J. L. Ocenasek. Experimental study: Hypergraph partitioning based on the simple and advanced algorithms BMDA and BOA. In *Proceedings of the Fifth International Conference on Soft Computing*, pages 124–130, 1999. Brno, Czech Republic.

[90] M. P. Servais, G. de Jaer, and J. R. Greene. Function optimization using multiple-base population based incremental learning. In *Proceedings of the*

Eight South African Workshop on Pattern Recognition, 1997.

[91] B. Sierra, E. Jiménez, I. Inza, P. Larrañaga, and J. Muruzábal. Rule induction using Estimation of Distribution Algorithms. In P. Larrañaga and J. A. Lozano, editors, *Estimation of Distribution Algorithms. A New Tool for Evolutionary Computation*. Kluwer Academic Publishers, 2001.

[92] M. Soto, A. Ochoa, S. Acid, and L. M. de Campos. Introducing the polytree aproximation of distribution algorithm. In *Second Symposium on Artificial Intelligence. Adaptive Systems. CIMAF 99*, pages 360–367, 1999. La Habana.

[93] R. Sukthankar, S. Baluja, and J. Hancock. Evolving an intelligent vehicle for tactical reasoning in traffic. In *International Conference on Robotics and Automation*, 1997.

[94] G. Syswerda. Simulated crossover in genetic algorithms. *Foundations of Genetic Algorithms 2*, pages 239–255, 1993.

[95] M. Thathachar and P. S. Sastry. Learning optimal discriminant functions through a cooperative game of automata. *IEEE Transactions on Systems, Man, and Cybernetics*, 17(1), 1987.

[96] C. H. M. van Kemenade. Building block filtering and mixing. In *Proceedings of the 1998 International Conference on Evolutionary Computation*. IEEE Press, 1998.

[97] B.-T. Zhang. A Bayesian framework for evolutionary computation. In *Proceedings of the Congress on Evolutionary Computation (CEC99), IEEE Press*, pages 722–727, 1999.

[98] B.-T. Zhang and D.-Y. Cho. Evolving neural trees for time series prediction using Bayesian evolutionary algorithms. In *Proceedings of the First IEEE Workshop on Combinations of Evolutionary Computation and Neural Networks (ECNN-2000)*, 2000.

[99] B.-T. Zhang and S.-Y. Shin. Bayesian evolutionary optimization using Helmholtz machines. In M. Schoenauer, K. Deb, G. Rudolph, X. Yao, E. Lutton, J. J. Merelo, and H.-P. Schwefel, editors, *Lecture Notes in Computer Science, 1917. Parallel Problem Solving from Nature – PPSN VI*, pages 827–836, 2000.

[100] Q. Zhang and H. Mühlenbein. On global convergence of FDA with proportionate selection. In *Second Symposium on Artificial Intelligence. Adaptive Systems. CIMAF 99*, pages 340–343, 1999. La Habana.

[101] A. A. Zhigljavsky. *Theory of Global Random Search*. Kluwer Academic Publishers, 1991.