Tema 3. Algoritmos de Estimación de Distribuciones

Abdelmalik Moujahid, Iñaki Inza y Pedro Larrañaga
Departamento de Ciencias de la Computación e Inteligencia Artificial
Universidad del País Vasco–Euskal Herriko Unibertsitatea

3.1 Introducción

El comportamiento de los algoritmos genéticos depende de un buen número de parámetros: los operadores de cruce y mutación, las probabilidades de cruce y mutación, el tamaño de la población, la tasa de reproducción generacional, el número de generaciones, etc. De hecho la determinación de valores adecuados para dichos parámetros constituye por si mismo un verdadearo problema de optimización. Por otra parte una mala elección de los valores de los parámetros puede llevar a que el algoritmo obtenga soluciones alejadas del óptimo. Este es uno de los motivos por los que desde hace varios años se han venido estudiando alternativas a los metodos heurísticos estocásticos existentes que no necesitasen el ajustar un número alto de parámetros.

Otro motivo básico por el que se ha desarrollado la búsqueda de nuevos heurísticos de optimización es por la necesidad de identificar las interrelaciones entre las variables utilizadas para representar a los individuos con la codificación utilizada.

Todo ello ha motivado el nacimiento de un nuevo método de búsqueda conocido como Algoritmo de Estimación de Distribuciones, que denotaremos por EDAs (*Estimation of Distribution Algorithms*).

Los EDAs son algoritmos heurísticos de optimización que basan su búsqueda –al igual que los algoritmos genéticos– en el caracter estocástico de la misma. Tambien al igual que los algoritmos genéticos los EDAs están basados en poblaciones que evolucionan. Sin embargo a diferencia de los algoritmos genéticos en los EDAs la evolución de las poblaciones no se lleva a cabo por medio de los operadores de cruce y mutación. En lugar de ello la nueva población de individuos se muestrea de una distribución de probabilidad, la cual es estimada de la base de datos conteniendo al conjunto de individuos seleccionados de entre los que constituyen la generación anterior.

Mientras que en los algoritmos genéticos las interrelaciones entre las variables representando a los individuos se tienen en cuenta de manera implícita, en los EDAs dichas interrelaciones se expresan de manera explícita a través de la distribución de probabilidad asociada con los individuos seleccionados en cada generación. De hecho la estimación de dicha distribución de probabilidad conjunta asociada a los individuos seleccionados en cada generación es la principal dificultad de esta aproximación.

En este capítulo vamos a presentar una aproximación general a los EDAs, es decir

una aproximación en la cual la distribución de probabilidad con la cual se modeliza el comportamiento de los individuos seleccionados en cada generación es genérica, si bien posteriormente particularizaremos para el caso más simple en el cual la distribución de probabilidad conjunta es factorizada como producto de distribuciones marginales univariantes.

3.2 Ilustrando EDAs por medio de un ejemplo

Con el objetivo de entender el funcionamiento de los diferentes componentes y pasos de los EDAs, aplicaremos la versión más simple de esta aproximación a un ejemplo muy sencillo de optimización.

	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

Tabla 3.1: La población inicial, D_0

Supongamos que tratamos de maximizar la función *OneMax* definida en un espacio de dimensión 6. Es decir, tratamos 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 por medio del muestreo de 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 conjunta de la cual se muestrea, se encuentra factorizada como un producto de seis distribuciones marginales univariantes, cada una de las cuales sigue un modelo de Bernouilli con parámetro igual a 0.5. Denotamos por D_0 el fichero conteniendo 20 casos –ver Tabla 3.1– obtenida por medio de esta simulación.

En un segundo paso, seleccionamos algunos de los individuos de D_0. Esto puede hacerse usando uno de los métodos de selección estandard en computación evolutiva. Supongamos que nuestro método de selección es truncación, y que seleccionamos a la mitad de la población. Denotamos por D_0^{Se} el fichero de casos conteniendo los indiviuos seleccionados. En caso de que existan empates en la función de evaluación de algunos individuos (situación que se verifica en los individuos numerados como 1, 9, 12, 18 y 20) la selección se lleva a cabo de manera probabilística entre los mismos. Por ejemplo, en este caso se necesitan seleccionar 3 individuos de entre el conjunto de individuos cuya función de evaluación vale 3.

	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

Tabla 3.2: Individuos seleccionados, D_0^{Se}, a partir de la población inicial

Una vez que tenemos seleccionados 10 individuos –véase la Tabla 3.2– nos interesa expresar explícitamente –por medio de la distribución de probabilidad conjunta– las características de los individuos seleccionados. Aunque somos conscientes de que sería interesante que dicha distribución de probabilidad conjunta tuviera en cuenta las interdependencias entre las variables, en este caso utilizaremos el modelo más simple posible para expresar dicha distribución de probabilidad conjunta. En dicho modelo, cada variable se va a considerar independiente del resto. Esto se expresa matemáticamente por medio de:

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

Es decir que tan sólo necesitamos 6 parámetros para especificar el modelo. Cada uno de dichos parámetros, $p(x_i|D_0^{Se})$ con $i = 1, ..., 6$, será estimado a partir del fichero de casos D_0^{Se} por medio de la frecuencia relativa correspondiente, $\hat{p}(X_i = 1|D_0^{Se})$.

En este ejemplo los valores de los parámetros resultan ser:

$$\begin{array}{lll} \hat{p}(X_1 = 1|D_0^{Se}) = 0{,}7 & \hat{p}(X_2 = 1|D_0^{Se}) = 0{,}7 & \hat{p}(X_3 = 1|D_0^{Se}) = 0{,}6 \\ \hat{p}(X_4 = 1|D_0^{Se}) = 0{,}6 & \hat{p}(X_5 = 1|D_0^{Se}) = 0{,}8 & \hat{p}(X_6 = 1|D_0^{Se}) = 0{,}7. \end{array} \tag{2}$$

Muestreando la distribución de probabilidad, $p_1(\boldsymbol{x})$, obtenemos una nueva población de individuos, D_1. La Tabla 3.3 representa el fichero de casos consistente en los 20 individuos obtenidos por medio de la simulación de $p_1(\boldsymbol{x})$. En dicha tabla se indica asimismo para cada individuo su valor de $h(\boldsymbol{x})$ asociado.

	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

Tabla 3.3: La primera generación de individuos: D_1

De nuevo seleccionamos 10 individuos de D_1, obteniendo –como puede verse en la Tabla 7.4– el fichero D_1^{Se}.

	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

Tabla 3.4: D_1^{Se}: Individuos seleccionados de la primera generación de inidviduos

A partir de este fichero de casos obtenemos:

$$p_2(\boldsymbol{x}) = p_2(x_1, \ldots, x_6) = \prod_{i=1}^{6} p(x_i | D_1^{Se}) \tag{3}$$

donde $p(x_i|D_1^{Se})$ con $i = 1, \ldots, 6$ se estima de D_1^{Se} por medio de su correspondiente frecuencia relativa, $\hat{p}(X_i = 1|D_1^{Se})$.

Como puede comprobarse, en este caso los valores de los parámetros son:

$$\begin{array}{lll} \hat{p}(X_1 = 1|D_1^{Se}) = 0{,}9 & \hat{p}(X_2 = 1|D_1^{Se}) = 0{,}8 & \hat{p}(X_3 = 1|D_1^{Se}) = 0{,}8 \\ \hat{p}(X_4 = 1|D_1^{Se}) = 0{,}7 & \hat{p}(X_5 = 1|D_1^{Se}) = 0{,}8 & \hat{p}(X_6 = 1|D_1^{Se}) = 0{,}7. \end{array} \quad (4)$$

Los pasos anteriores se repiten hasta que se verifique una determinada condición de parada.

3.3. Aproximación general

En este apartado vamos a generalizar lo presentado en el apartado anterior. Tal y como se ha visto, la aproximación EDA consiste en un heurístico probabilístico de búsqueda, basado en poblaciones que evolucionan, y fundamentado en tres pasos básicos a iterar.

Estos tres pasos consisten en:

1. seleccionar algunos individuos de la población
2. estimar el modelo probabilístico subyacente a dichos individuos seleccionados
3. muestrear de la distribución de probabilidad aprendida, con objeto de obtener una nueva población de individuos

y son repetidos hasta que se verifique un criterio de parada, previamente establecido.

En la Figura 3.1, al igual que en el pseudocódigo de la Figura 3.2, podemos ver un esquema de la aproximación EDA. Partimos de M individuos generados al azar, en nuestro ejemplo anterior por medio de una distribución uniforme para cada variable. Estos M individuos constituyen la población inicial, D_0. A continuación, cada uno de dichos individuos es evaluado. En un primer paso, un número $N(N \leq M)$ de individuos es seleccionado (habitualmente aquellos con mejor función objetivo). En un segundo paso se lleva a cabo la inducción del modelo probabilístico n-dimensional que mejor refleja las interdependencias entre las n variables. En un tercer paso, M nuevos individuos (la nueva población) se obtienen por medio de la simulación de la distribución de probabilidad aprendida en el paso anterior. Estos tres pasos se repiten hasta que se verifique una condición de parada. Ejemplos de condiciones de parada son: determinación de un número máximo de poblaciones, o de un número máximo de individuos evaluados, uniformidad en la población generada, no mejora con respecto al mejor de los individuos obtenidos en las generaciones previas, etc.

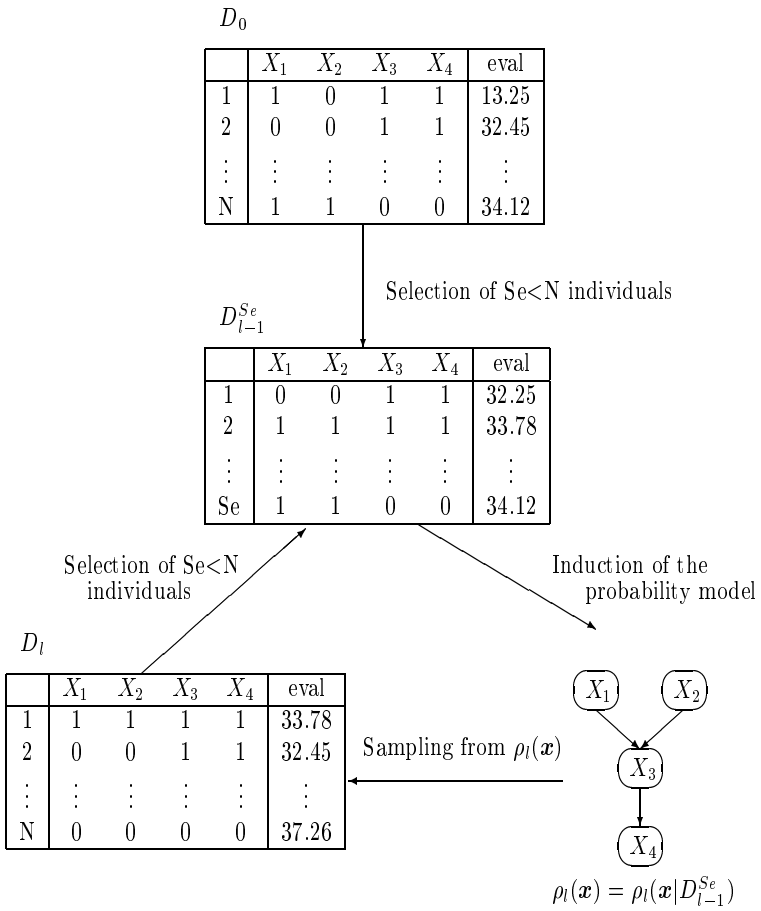

Figura 1: Figura 3.1: Ilustración de la aproximación EDA a la optimización

---

EDA

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

Repeat 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} de acorde con el metodo de seleccion

$p_l(\boldsymbol{x}) = p(\boldsymbol{x}|D_{l-1}^{Se}) \leftarrow$ Estimar la distribucion de probabilidad de que un individuo se encuentre en los individuos seleccionados

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

Figura 3.2: Pseudocódigo de la aproximación EDA

El mayor problema con los EDAs es como estimar la distribución de probabilidad $p_l(\boldsymbol{x})$. Obviamente la computación de todos los parámetros necesarios para especificar la distribución de probabilidad conjunta no es práctica. Este problema trae como consecuencia la aproximación de la distribución de probabilidad conjunta por medio de distintas factorizaciones –más o menos complejas– de la misma.

3.4 La aproximación UMDA

Tal y como puede verse en el pseudocódigo de la Figura 3.3, el modelo probabilístico utilizado por el algoritmo UMDA (*Univariate Marginal Distribution Algorithm*) es el más simple posible, y coincide con el que nos ha servido para ilustrar el ejemplo del apartado anterior. En concreto, en cada generación la distribución de probabilidad conjunta, $p_l(\boldsymbol{x})$, que sirve para estimar el comportamiento de los individuos seleccionados, se factoriza como un producto de distribuciones marginales univariantes e independientes. Es decir:

$$p_l(\boldsymbol{x}) = p(\boldsymbol{x}|D_{l-1}^{Se}) = \prod_{i=1}^{n} p_l(x_i). \tag{5}$$

UMDA

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

Repeat 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} de acorde al metodo de seleccion

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

$D_l \leftarrow$ Muestrar M individuos (la nueva poblacion) de $p_l(\boldsymbol{x})$

Figura 3.3: Pseudocódigo del algoritmo UMDA

Cada distribución marginal se estima a partir de las frecuencias marginales:

$$p_l(x_i) = \frac{\sum_{j=1}^{N} \delta_j(X_i = x_i|D_{l-1}^{Se})}{N} \tag{6}$$

donde

$$\delta_j(X_i = x_i|D_{l-1}^{Se}) = \begin{cases} 1 & \text{si en el } j\text{-ésimo caso de } D_{l-1}^{Se},\ X_i = x_i \\ 0 & \text{en otro caso.} \end{cases} \tag{7}$$

3.5 La aproximación 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.

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

while no convergencia **do**
begin

Usando $p_l(\boldsymbol{x})$ obtener M individuos: $\boldsymbol{x}_1^l, \ldots, \boldsymbol{x}_k^l, \ldots, \boldsymbol{x}_M^l$

Evaluar y ordenar $\boldsymbol{x}_1^l, \ldots, \boldsymbol{x}_k^l, \ldots, \boldsymbol{x}_M^l$

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

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**
$p_{l+1}(x_i) =$
$(1-\alpha)p_l(x_i) + \alpha\frac{1}{N}\sum_{k=1}^{N} x_{i,k:M}^l$

end

Figura 3.4: Pseudocódigo del algoritmo PBIL

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:

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

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

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

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 3.4 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.

3.6 La aproximación cGA

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

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: $\boldsymbol{x}_1^l, \boldsymbol{x}_2^l$

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 3.5: Pseudocódigo del algoritmo cGA

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 3.5 muestra un pseudocódigo para el cGA.

3.7 Otras aproximaciones más complejas

En problemas de optimización que surgen en el mundo real la aproximación anterior (UMDA) no es muy adecuada, ya que supone que las variables no interactúan entre si. Esta falta de interdependencia entre las variables hace que el modelo UMDA esté muy alejado de lo que en realidad ocurre. Es por ello por lo que se han desarrollado algoritmos, que perteneciendo a la familia de EDAs, incorporan la posibilidad de tener en cuenta dichas interrelaciones entre las variables. Para ello se hace necesario utilizar una estimación de la distribución de probabilidad conjunta que no consista simplemente en el producto de las distribuciones de probabilidad marginales univariantes, sino que utilice modelos más complejos.

Aumentando en complejidad, se pueden considerar modelos que tengan en cuenta estadísticos de orden dos o incluso de orden superior. En realidad la aproximación más genérica se obtiene cuando en cada generación la distribución de probabilidad se factoriza por medio de redes Bayesianas (optimización combinatorial) o de redes Gaussianas (optimización en dominios continuos). Queda sin embargo, lejos del

objetivo de estos apuntes el presentar ejemplos de tales aproximaciones.

Referencias

1. S. Baluja (1994). Population–based incremental learning: A method for integrating genetic based search function optimization and competitive learning. *Technical Report CMU–CS–94–163. Carnegie Mellon University.*

2. S. Baluja, R. Caruana (1995). Removing the genetic from standard genetic algorithm. *Proceedings of the International Conference on Machine Learning*, 38–46.

3. R. Etxeberria, P. Larrañaga (1999). Global optimization with Bayesian networks. *II Symposium on Artificial Intelligence. CIMAF99. Special Session on Distributions and Evolutionary Optimization*, 332–339.

4. G. Harik, F. Lobo, D. Goldberg (1998). The compact genetic algorithm. *Proceedings of the IEEE Conference on Evolutionary Computation*, 523–528.

5. P. Larrañaga, R. Etxeberria, J. A. Lozano, J. M. Peña (2000a). Combinatorial optimization by learning and simulation of Bayesian networks. *Proceedings of the Sixteenth Conference on Uncertainty in Artificial Intelligence*, 343–352.

6. P. Larrañaga, R. Etxeberria, J. A. Lozano, J. M. Peña (2000b). Optimization in continuous domains by learning and simulation of Gaussian networks. *Proceedings of the 2000 Genetic and Evolutionary Computation Conference Workshop Program*, 201–204.

7. P. Larrañaga, J. A. Lozano (2001). *Estimation of Distribution Algorithms. A New Tool for Evolutionary Computation.* Kluwer Academic Publishers.

8. H. Mühlenbein (1998). The Equation for Response to Selection and its Use for Prediction. *Evolutionary Computation*, **5**, 303–346.

9. H. Mühlenbein, G. Paaß(1996). From Recombination of Genes to the Estimation of Distributions I. Binary Parameters. *Lecture Notes in Computer Science 1411: Parallel Problem Solving from Nature - PPSN IV*, 178–187.

10. M. Pelikan, D. E. Goldberg, E. Cantú-Paz (1999). BOA: The Bayesian optimization algorithm. *Proceedings of the Genetic and Evolutionary Computation Conference GECCO-99*, 525–532.