Una breve introducción a los Algoritmos Evolutivos con Estimación de Distribuciones

Alberto Ochoa
Instituto de Cibernética Matemática y Física
Habana. Cuba
ochoa@icmf.inf.cu

EDA

Algoritmo con Estimación de Distribuciones

Algoritmo 1 EDA.

Paso 1	$t \leftarrow 1$. Generar N puntos aleatoriamente.
Paso 2	Evaluar los puntos en $f(x)$. Seleccionar $\bar{N}$ puntos y crear el conjunto seleccionado S.
Paso 3	Estimar la distribución del conjunto seleccionado $p^s(x,t)$ a partir de S.
Paso 4	Generar N nuevos puntos de acuerdo a $p(x,t+1) \approx p^s(x,t)$.
Paso 5	$t \leftarrow t+1$. Si no se cumple el criterio de parada ir al paso 2.

Simple Genetic Algorithm

---

1 $t \Leftarrow 0$. Generate $N \gg 0$ points by uniform distribution.

2 **do** {

3 Select $M \leq N$ points.

4 Generate N new points, performing crossover and mutation from the selected set.

5 $t \Leftarrow t + 1$

6 } **until** Termination criterion reached.

- Los EDA se introducen como algoritmos evolutivos que construyen **DISTRIBUCIONES de BÚSQUEDA** sobre el ¨genotipo¨ del problema de optimización

¿Qué es una distribución de Búsqueda?

- Una distribución de probabilidades que se define sobre el espacio de todas las posibles configuraciones de un problema de optimización.
- Idealmente esta asigna probabilidad máxima al óptimo (óptimos) y señala un ¨camino evolutivo¨ de máxima probabilidad local que orienta la búsqueda.

Distribuciones de Búsqueda

- **Engañosas** - asigna mayor probabilidad a los peores individuos global y localmente.
- **Inútiles** - asigna la misma probabilidad a todas las configuraciones del espacio. (PASEO ALEATORIO).
- **Buenas** - nos dan alguna información, que usada inteligentemente, hace más eficiente la búsqueda.

El Modelo 1 es una aproximación INÚTIL !!!

- Modelo 1
  $p(x1,x2)=p(x1)p(x2)$
- Modelo exacto
  $p(x1,x2)=p(x1)p(x2|x1)$

- $p(x1=0)=0.5$
- $p(x2=0)=0.5$
- $p(x2=0|x1=0)=0.75$
- $p(x2=1|x1=0)=0.25$
- $p(x2=0|x1=1)=0.25$
- $p(x2=1|x1=1)=0.75$

Epistasis

- Correlación
- Independencia/Dependencia estadística
- Dependencias no-lineales

Encontrar buenas distribuciones de búsqueda para problemas con alto grado de epistasis no es una tarea fácil.

EL AG trabaja a ¨ciegas¨ con distribuciones de búsqueda

- Todo conjunto es una muestra de al menos una distribución.
- El conjunto de individuos generados por el AG tiene distribución **Dg**.
- El conjunto de individuos seleccionados por el AG tiene distribución **Ds**.

¿Dg = Ds ?

¿Cómo construir Distribuciones de búsqueda?

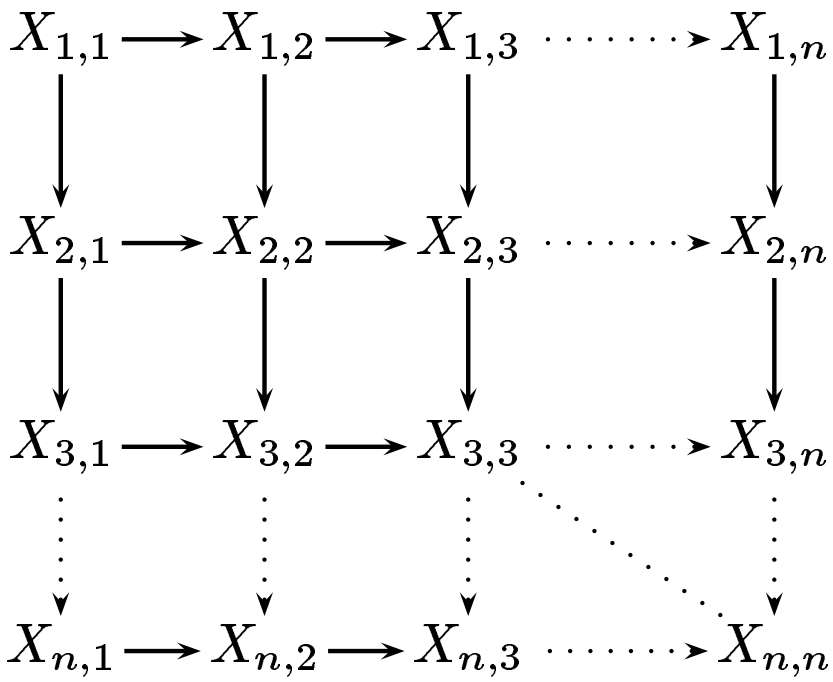
- Primer problema - complejidad exponencial
- p(x1,..,x300) necesita 2^300-1 parámetros
- Si sabemos que:

$$p(x1,...,x300)=p(x1,x2,x3)*...*p(x4,x5,x6)*\\ p(x298,x299,x300)$$

entonces se necesitan

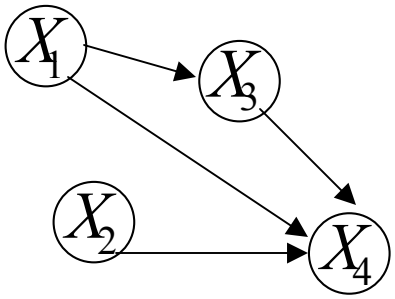
$$100*(2^3-1)=700 \text{ parámetros}$$

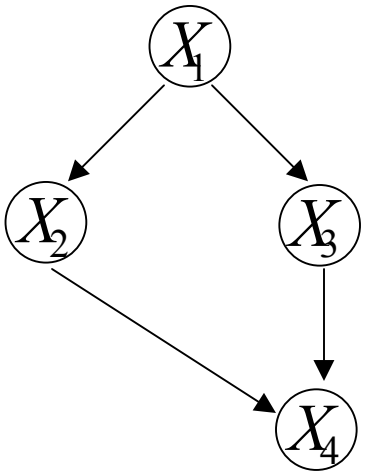
Factorization of the grid

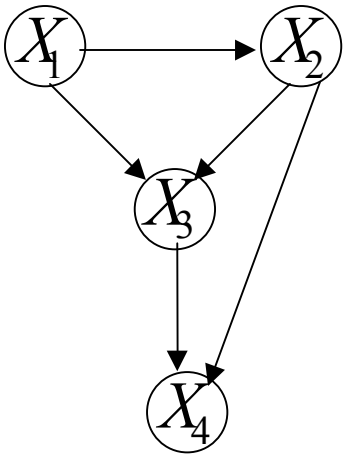


Redes Bayesianas multiconectadas

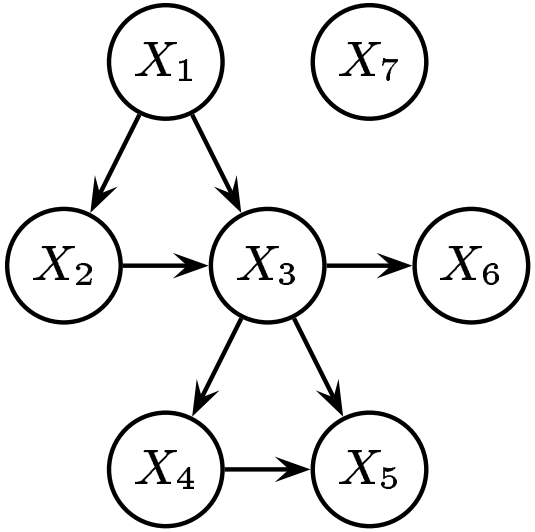
$$p(x_1, x_{2,\ldots,} x_n) = \prod_{i=1}^{n} p(x_i \mid \pi_{x_i})$$







Example of a Bayesian Network



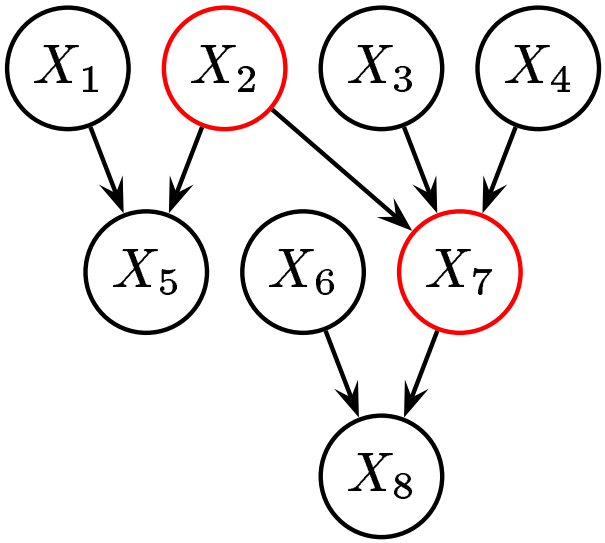
Induces the following factorisation:

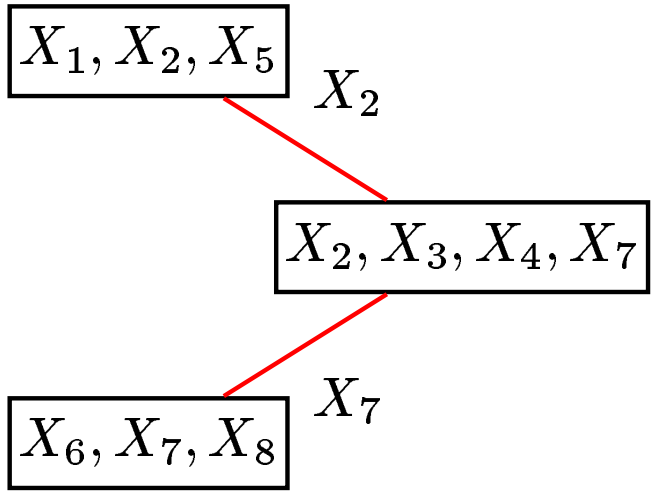
$$p(x) = p(x_1)p(x_2|x_1)p(x_3|x_1, x_2)p(x_4|x_3)p(x_5|x_3, x_4)p(x_6|x_3)p(x_7)$$

"Probabilistic Logic Sampling" (unlucky name)

Junction Trees

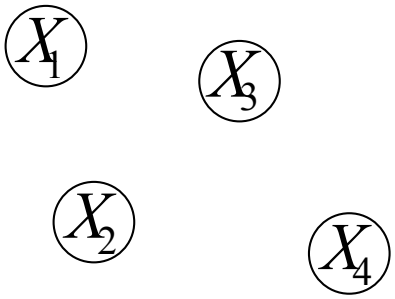
- Structure equivalent to Bayesian Networks
- Codes factorization of distribution
- For Polytrees: Child and its parents form one node



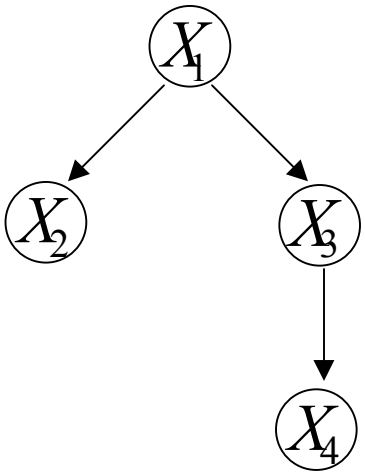


Redes Bayesianas simplemente conectadas o poliárboles

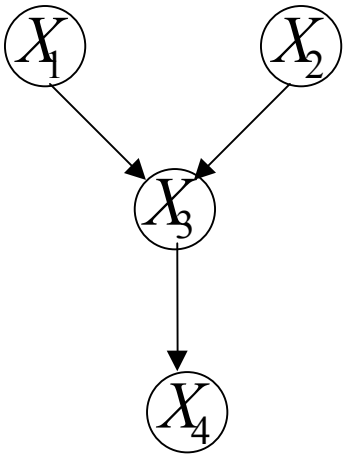
$$p(x_1, x_{2,\ldots,} x_n) = \prod_{i=1}^{n} p(x_i | \pi_{x_i})$$



Grafo totalmente desconectado



Arbol



Poliárbol

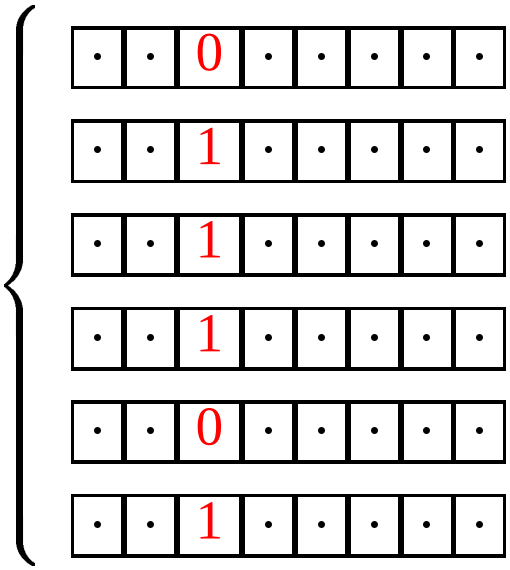
Crossover – The Genotype Perspective

Uniform Crossover: Every bits is chosen randomly from two parents.

Parent 1:	1	1	0	0	1	1	0	1	0	1	0	1
Parent 2:	1	0	1	0	0	1	0	0	1	0	1	1
Child:	1	1	1	0	0	1	0	1	1	1	0	1

Crossover – The Gene Perspective

Possible parents



$\Rightarrow$ Child has a 60% chance of having bit 3 set to 1 !

UMDA: Univariate Marginal Distribution Algorithm

---

1 $t \Leftarrow 0$. Generate $N \gg 0$ points by uniform distribution.
2 **do** {
3 Select $M \leq N$ points. Estimate bit probabilities $p_i(X_i, t)$.
4 Generate N points according to $p(x, t+1) = \prod_i p_i(x_i, t)$.
5 $t \Leftarrow t + 1$
6 } **until** Termination criterion reached.

UMDA *approximates* the simple genetic algorithm *SGA*.

The product distribution

The product of the simple univariate marginal frequencies $p_i(X_i, t)$ defines the product distribution

$$\tilde{p}(x_1, \ldots, x_n, t) = \prod_i p_i(x_i, t)$$

Corresponds to *Mean Field* theory.

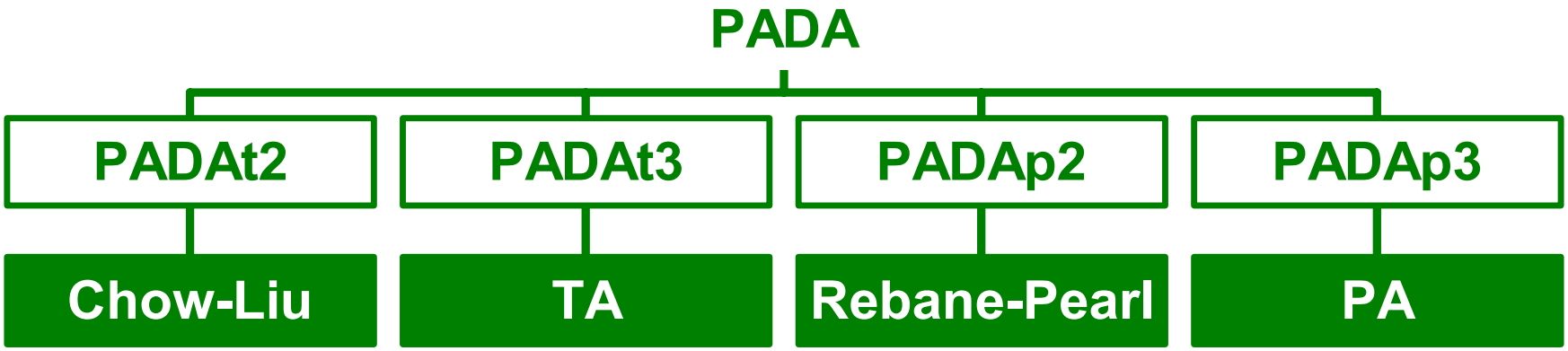
Associated to distribution and fitness function is the **average fitness**

$$W_f(p) = \sum_x p(x, t) f(x)$$

For *UMDA*, the average fitness is a function of the $p_i(X_i, t)$.

PADA

Algoritmo con Distribución basada en Poliárbol



¡Usar solamente marginales de hasta tercer orden!

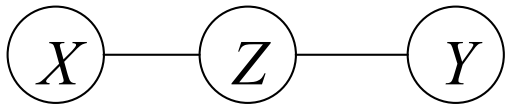
PADA = EDA + LPA

Esquema de un algoritmo de aprendizaje para poliárboles

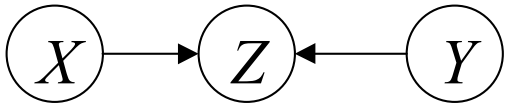
Construcción del esqueleto

- Comenzar con un grafo sin aristas.
- Insertar las aristas de mayor dependencia $\begin{cases} I(X,Y) \\ I(X,Y|Z) \end{cases}$

Orientación de los patrones cabeza-cabeza.

- Si 

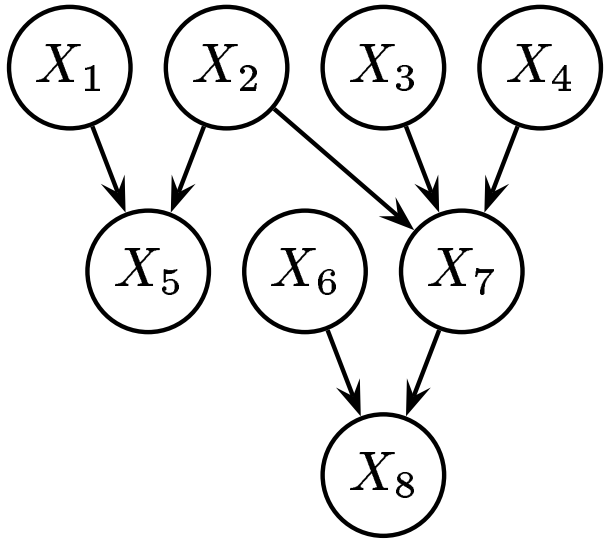
y

- Si $\begin{cases} I(X,Y) = 0 \\ I(X,Y|Z) > I(X,Y) \end{cases}$ entonces 

Orientación de las aristas restantes

Polytrees

- Polytree: Singly connected Bayesian Network
- But direction of edges is not restricted, therefore many roots are allowed
- At most $n - 1$ edges possible
- Expected number of parents ≈ 1



PADA – Polytree Approximation Distribution Algorithm

---

1 $t \Leftarrow 0$. Generate $N \gg 0$ points by uniform distribution.
2 **do** {
3 Selection of promising points.
4 Construct Polytree from points.
5 Compute the conditional probabilities $p^s(x_{b_i}|x_{c_i}, t)$.
6 Generate a new population according to
$p(x, t+1) = \prod_{i=1}^{m} p^s(x_{b_i}|x_{c_i}, t)$.
7 $t \Leftarrow t + 1$
8 } **until** Termination criterion reached.

PADA2: *PADA* with Bivariate Marginals

- For all pairs of variables, calculate their *mutual information*

$$I(X;Y) = \sum_{x,y} p(x,y) \log \frac{p(x,y)}{p(x)p(y)}$$
$$= H(X) + H(Y) - H(X,Y)$$

- Build a *Maximum Spanning Tree* of the variables
- For $X - Z - Y$ structure, if $I(X;Y) = 0$, direct edged towards Z
- Direct other edges randomly, avoiding new $X \rightarrow Z \leftarrow Y$ patterns
- Perform *FDA* with this Bayesian Network

PADA2: The Sampling Problem

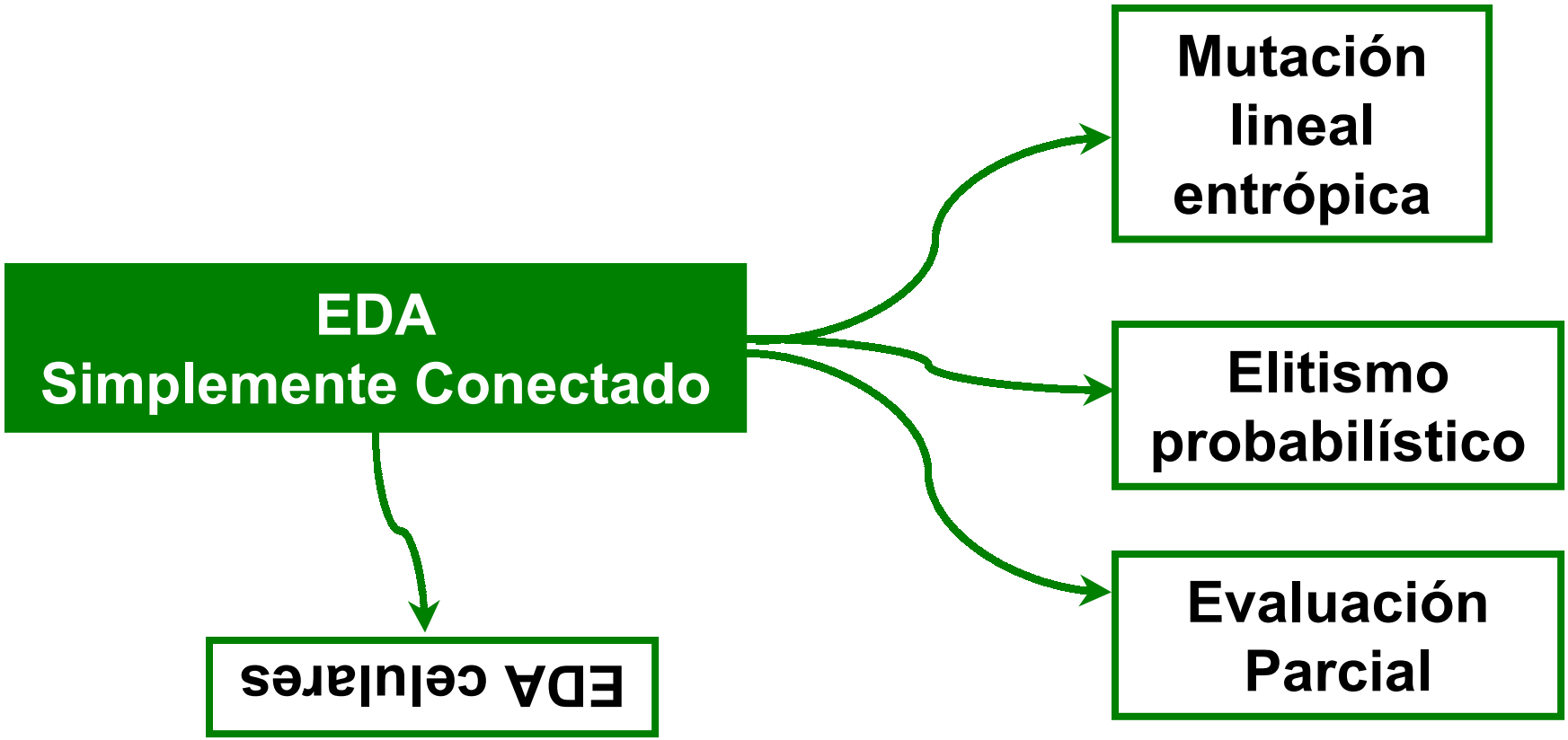
- *PADA* needs higher order marginals for sampling
- Costly calculation of these marginals
- Needs bigger population size for good estimation

  (particularly bad in pattern recognition, where fitness evaluation often is expensive)

$\Rightarrow$ Solution: Calculate higher-order marginals from (already calculated) bivariates!

But how to choose a distribution consistent with the given bivariates?

En búsqueda de mecanismos alternativos que puedan potenciar los modelos simples sin perder su mayor ventaja.



Mutation and the *UMDA*

- Consider N tosses of a coin with m times head. *Maximum Likelihood* estimate is $Pr(\text{head}) = m/N$.
- Bayesian estimate yields $Pr(\text{head}) = (m + r)/(N + 2r)$ with *Hyper parameter* $r \geq 0$.
- This is equivalent to the classical *mutation* of genetic algorithms, where generated bits are flipped with a certain probability μ with $\mu = r/(N + 2r)$.

The algorithms get more robust by using some sort of mutation.

Choice of the hyperparameter

- The greater the hyperparameter r, the nearer the resulting distribution is to the uniform distribution.
- An upper bound to r can be calculated by requiring that a population of all optima regenerates these with a probability 30%.
- For *UMDA*, this leads to a *recommendation of* $r = N/n$, popsize by bit length.
- It turns out that *the recommended prior gives very good results for many problems*.
- This can also be extended to *FDA* and conditional probabilities.

Elitismo y Evolución

Elitismo de tamaño k

Copia los mejores individuos de la población *t* en la *t+1*.

Elitismo probabilístico

Genera los individuos más probables de las distribuciones $p(x,t)$ o $p^s(x,t)$ y los **copia** en la población *t+1*.

¿ Cómo generar los puntos elitistas probabilísticos?

El elitismo disminuye el número de evaluaciones

Efecto del elitismo con PADAp2 en Deceptive3.

Algoritmo	Función objetivo	*N*	*n*	τ	*Elit*	*ElitProb*	*G*	*%Éxito*	*Neval*
$PADA_{p2}$	Deceptive3	500	18	0.3	0	0	6.75±1.43	79	3375
					150	0	7.04±1.46	100	2614
					300	0	9.07±2.29	100	2114
					450	0	14.90±4.91	20	-
					300	5	8.05±1.60	96	1945
					300	10	7.36±1.68	98	1835

- Velocidad de convergencia y número de evaluaciones.
- “Memoria”.