

Computación Bioinspirada

Dr. Edward Hinojosa Cárdenas
ehinojosa@unsa.edu.pe

Programación Evolutiva

- Por razones históricas, la PE ha sido asociada durante mucho tiempo a tareas de predicción y al uso de MEF como su representación.
- A partir de los '90, las variantes de PE para optimización de vectores de parámetros reales se han vuelto más frecuentes
 - Estas variantes se han posicionado como variantes estándar de PE

Programación Evolutiva

- PE fue desarrollada originalmente para simular a la evolución como un proceso de aprendizaje
 - Con el objetivo de generar inteligencia artificial
- Inteligencia: capacidad de un sistema de adaptar su comportamiento para lograr objetivos predefinidos en un ambiente dado (comportamiento adaptado)
- La capacidad para predecir el ambiente fue considerada como un requisito para lograr adaptabilidad (comportamiento inteligente o adaptado)
- Por lo tanto, la capacidad para predecir el ambiente es clave para lograr inteligencia

Programación Evolutiva

- La PE se basa en la evolución natural, pero considerando que el proceso evolutivo no se centra en el nivel de los individuos, sino en el nivel de las especies enteras.
- En la PE cada individuo representa una especie, por lo tanto no existe intercambio a través genético a través de especies, por ello se omite el operador de cruzamiento.

Programación Evolutiva

- La PE enfatiza en la optimización de modelos de comportamiento:
 - Modelar el comportamiento con el fin de predecir lo que va a suceder (Predicción)
 - Capturar la interacción del sistema con el ambiente.

Maquinas de Estado Finito

- Una forma común de predecir una acción consiste en el análisis de acciones anteriores.
- En el contexto de una MEF, cada acción o suceso puede ser representada por un símbolo.
 - Dada una secuencia de símbolos, se debe predecir cual será el próximo símbolo.

Maquinas de Estado Finito

- Los símbolos deben pertenecer a un alfabeto finito.
- Para realizar la optimización debemos:
 - Analizar la secuencia de símbolos.
 - Generar una salida que optimice un función de aptitud dada, la cual envuelve el pronóstico del próximo símbolo de la secuencia:
 - Mercado Financiero, Predicción del Tiempo, etc.

Maquinas de Estado Finito

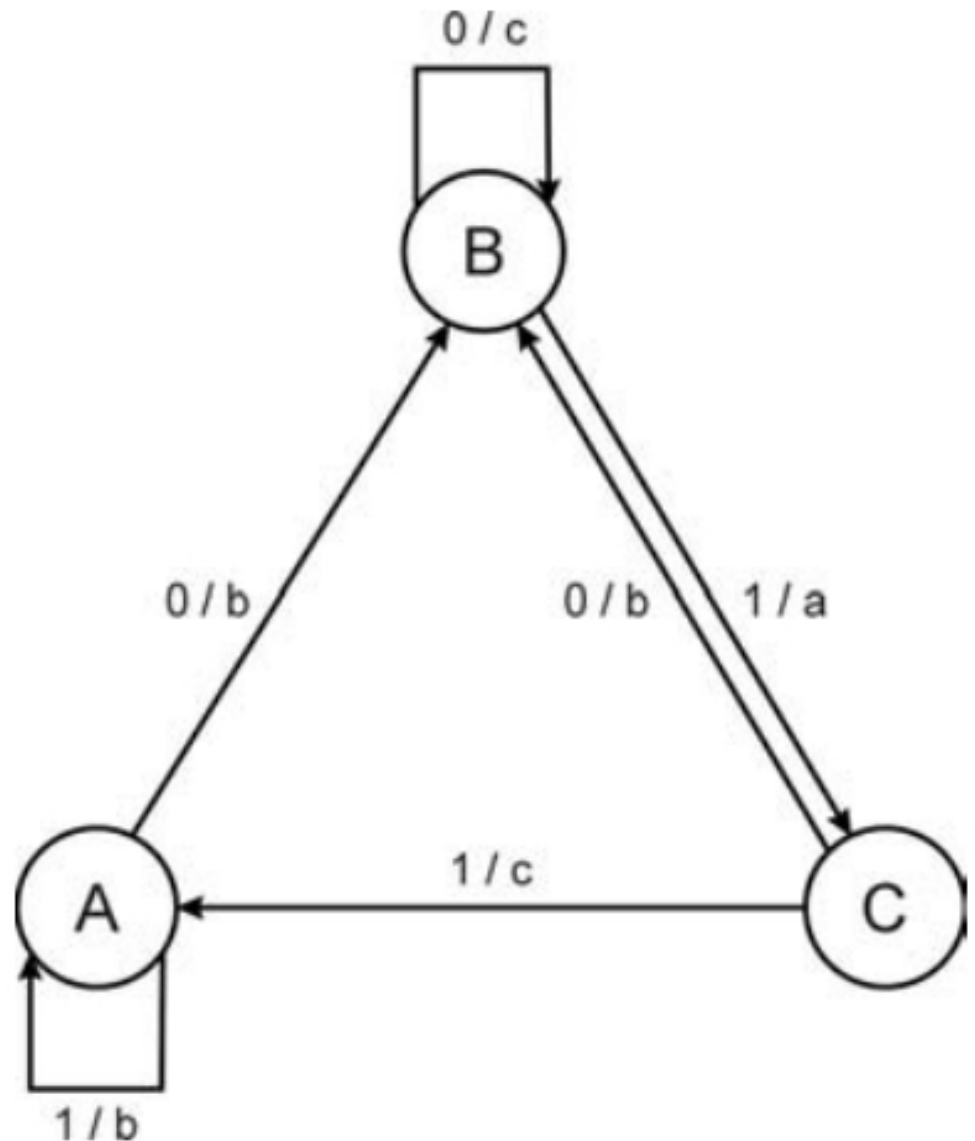
- Cuando se considera por un alfabeto finito de símbolos, responde con otro alfabeto finito de símbolos y posee un número finito de estados.
- Alfabetos de entrada y salida no son necesariamente idénticos.

Maquinas de Estado Finito

- Inicialmente en PE, los predictores fueron evolucionados en la forma de máquinas de estado finito
- Máquina de Estado Finito (MEF)
 - Estados (S)
 - Entradas (I)
 - Salidas (O)
 - Función de transición $\delta : S \times I \rightarrow S \times O$
 - Transforma una cadena de símbolos de entrada en una cadena de símbolos de salida
- Pueden ser usadas para predicciones (ej.: predecir el siguiente símbolo de entrada en una cadena de entrada).

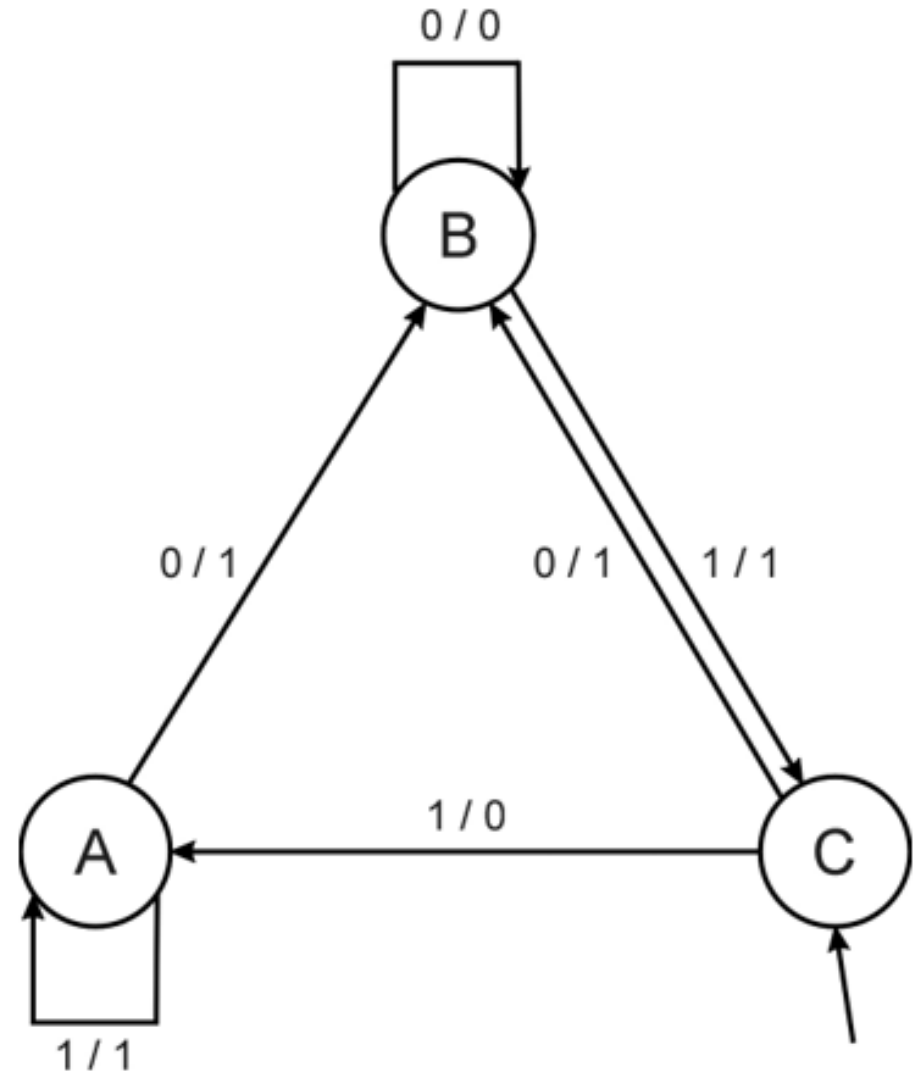
Ejemplo de MEF

- Considerar la MEF con:
 - $S = \{A, B, C\}$
 - $I = \{0, 1\}$
 - $O = \{a, b, c\}$
 - δ especificada por el diagrama.



Ejemplo de una MEF como un predictor


- Tarea: predecir la siguiente entrada
- Calidad de Pred. o Aptitud:
 $\text{in}(i+1) = \text{out}(i)$
- Estado inicial: C
- Secuencia de entrada:
011101
- Secuencia de salida:
110111
- Aptitud: 3 aciertos de 5



Programación Evolutiva

```
Procedure EP{
    t = 0;
    Initialize P(t);
    Evaluate P(t);
    While (Not Done)
    {
        Parents(t) = Select_Parents(P(t));
        Offspring(t) = Procreate(Parents(t));
        Evaluate(Offspring(t));
        P(t+1) = Select_Survivors(P(t), Offspring(t));
        t = t + 1;
    }
}
```

No existe cruzamiento,
solo Mutación

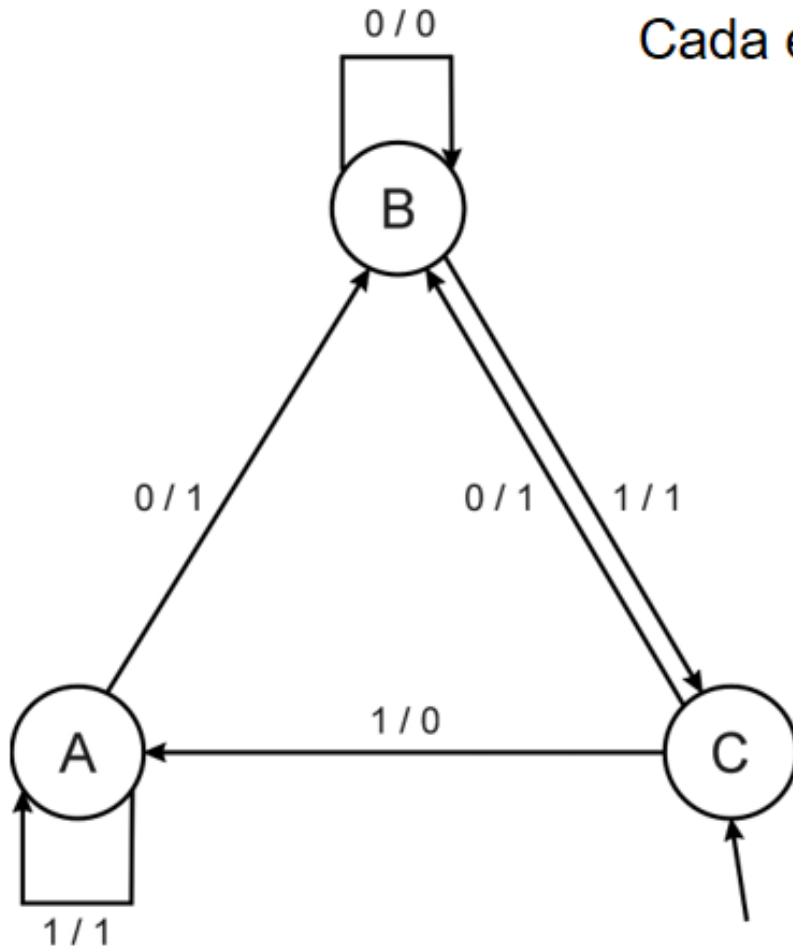


Programación Evolutiva: Codificación

- Aunque PE puede tener individuos de tamaño variable, es posible evolucionar una MEF con PE donde los individuos tienen tamaño fijo:
 - Definir un número máximo de estados.
- Por ejemplo, vamos a considerar la máquina de predicción presentada anteriormente, la cual puede tener como máximo 4 estados.

Programación Evolutiva: Codificación

Cada estado puede ser representado por 7 caracteres



| No. | Representación |
|-----|-----------------------|
| 0 | 1 activo; 0 no activo |
| 1 | símbolo de entrada |
| 2 | símbolo de entrada |
| 3 | símbolo de salida |
| 4 | símbolo de salida |
| 5 | estado de salida |
| 6 | estado de salida |

| A | B | C | D |
|---------------|---------------|---------------|---------------|
| 1 1 0 1 1 A B | 1 0 1 0 1 B C | 1 1 0 0 1 A B | 0 0 0 0 0 D A |

Programación Evolutiva: Codificación

- Como vimos, cada estado puede ser representado por una cadena de 7 caracteres:

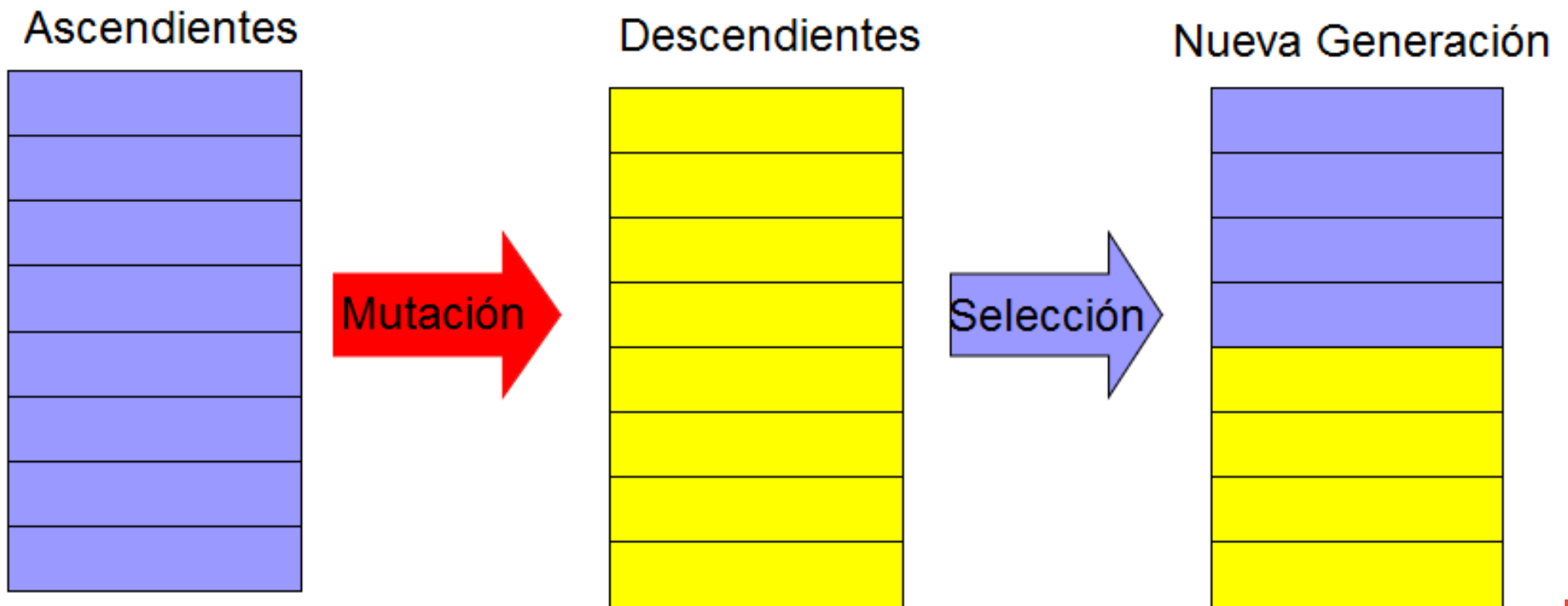
| A | B | C | D |
|---------------|---------------|---------------|---------------|
| 1 1 0 1 1 A B | 1 0 1 0 1 B C | 1 1 0 0 1 A B | 0 0 0 0 0 D A |

Operadores usados en la PE: Mutación

- Diferente a los AG donde el cruzamiento es un componente importante para la producción de una nueva generación, la mutación es el único operador usado en la PE.
- Podemos considerar, cada miembro de la población después de la mutación produce un hijo. Si tenemos M (tamaño de la población) ascendientes serán generados M descendientes.

Operadores usados en la PE: Mutación

- La mejor mitad de la población ascendiente y la mejor mitad de la población descendiente se unen para formar la nueva generación.



Operadores usados en la PE: Mutación

- Cinco tipos de mutación pueden ocurrir en una máquina de estados finitos. Se recomienda las siguientes probabilidades (siempre se realiza una mutación que genera un individuo **viable diferente al ascendente**):

| Valor | Acción |
|-----------|-------------------------------|
| 0.0 – 0.2 | Desactivar un estado |
| 0.2 – 0.4 | Cambiar estado inicial |
| 0.4 – 0.6 | Cambiar un símbolo de entrada |
| 0.6 – 0.8 | Cambiar un símbolo de salida |
| 0.8 – 1.0 | Activar un estado |

Operadores usados en la PE:

Mutación

- En muchos casos la mutación puede generar individuos inviables, como transiciones que no sean posibles (pues un estado puede haber sido eliminado).
- Esos problemas deben ser identificados y corregidos durante la implementación.

Criterios de Parada

- Se pueden considerar diferentes criterios de parada, por ejemplo:
 - Cuando el fitness es satisfactorio.
 - Definir un número de generaciones.

Programación Evolutiva

- Por razones históricas, la PE ha sido asociada durante mucho tiempo a tareas de predicción y al uso de MEF como su representación.
- A partir de los '90, las variantes de PE para optimización de vectores de parámetros reales se han vuelto más frecuentes
 - Estas variantes se han posicionado como variantes estándar de PE

Programación Evolutiva

- Para representación de vectores con valores reales, la programación evolutiva es muy similar a las estrategias evolutivas sin cruzamiento.
- Un método de selección típico es seleccionar a todos los individuos en la población para que sean μ ascendentes, para mutar a cada ascendente para generar μ descendientes.
- Después seleccionar los $\mu/2$ mejores ascendentes y $\mu/2$ descendientes para la siguiente población.

Programación Evolutiva

- PE aplica auto-adaptación de los parámetros de mutación.
- En esta variante un individuo sufre esta transformación:

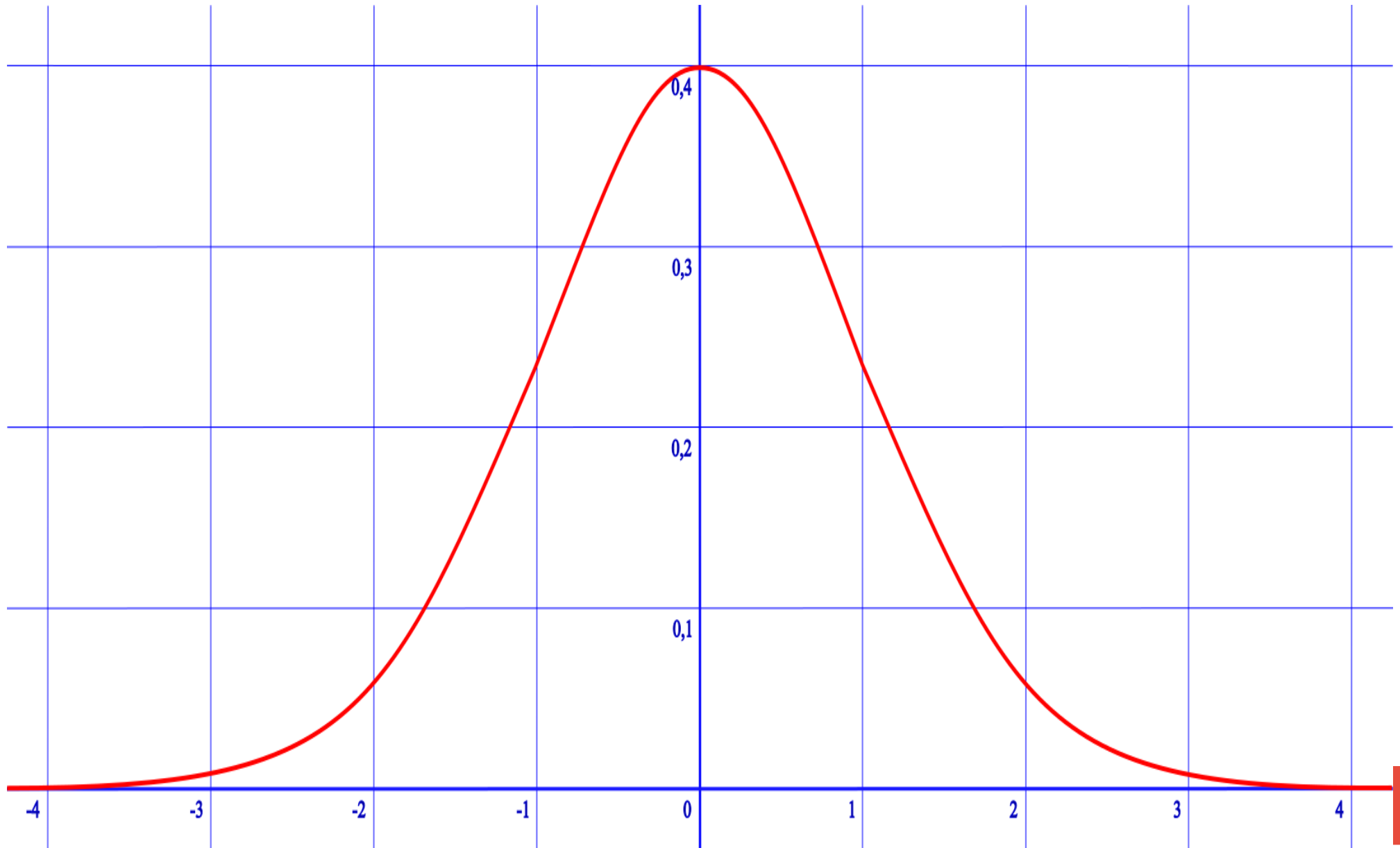
$$\langle x_1, \dots, x_n, \sigma_1, \dots, \sigma_n \rangle \quad \begin{aligned} \sigma'_i &= \sigma_i (1 + \alpha N(0,1)), \\ x'_i &= x_i + \sigma'_i N_i(0,1). \end{aligned} \quad \langle x'_1, \dots, x'_n, \sigma'_1, \dots, \sigma'_n \rangle$$
$$\alpha \approx 2$$

Programación Evolutiva

- Donde $N(0,1)$ es un vector de números Gaussianos independientes con una media de 0 y desviaciones estándar 1.
- La fórmula de una distribución normal está dada por:

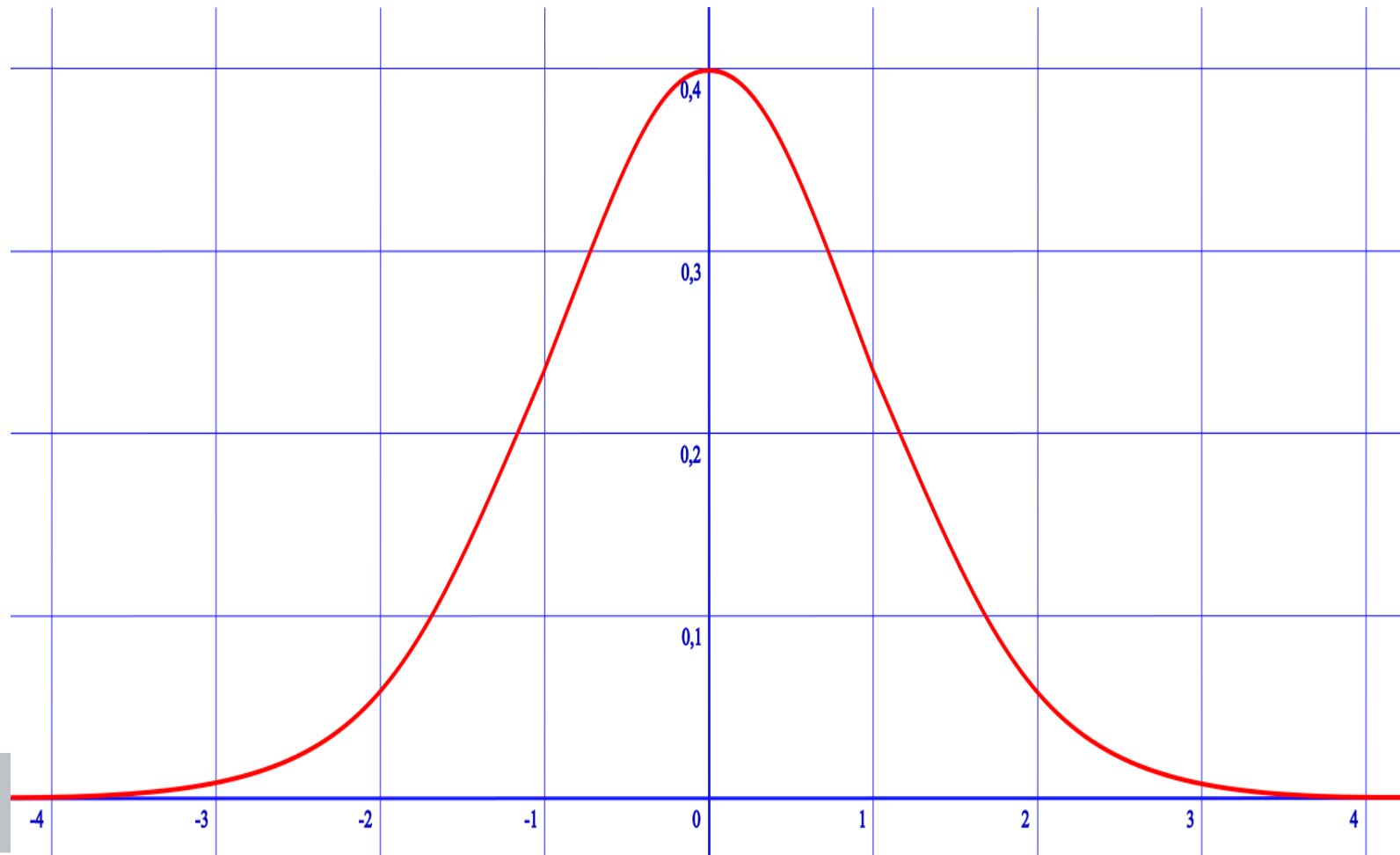
$$N(0, \sigma, x) = \frac{e^{-\frac{1}{2} \left(\frac{x}{\sigma} \right)^2}}{\sigma \sqrt{2\pi}}$$

Programación Evolutiva



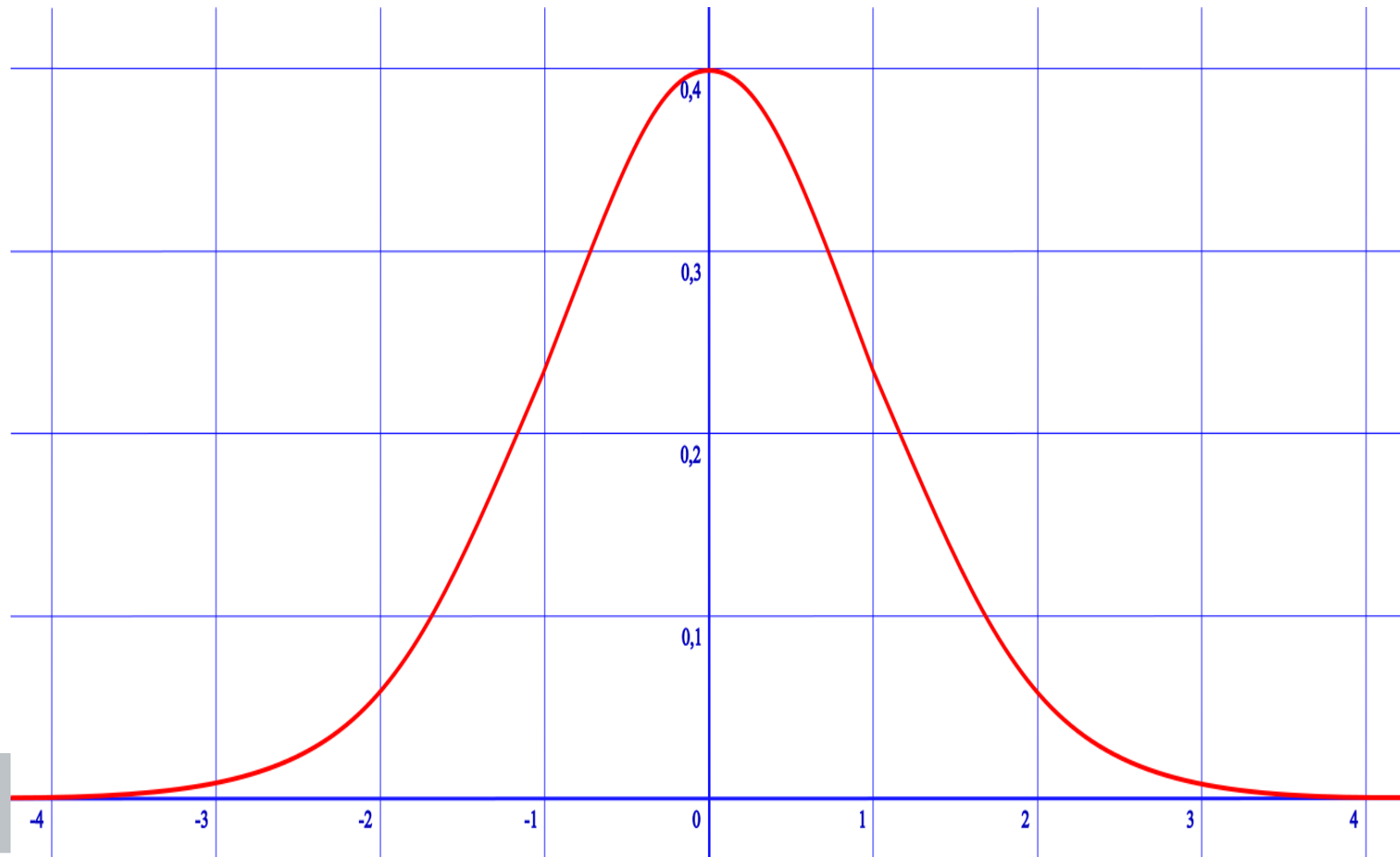
Programación Evolutiva

$$\frac{\left(e^{-\left(0.5 \times (\theta^2) \right)} \right)}{\sqrt{2 \times \pi}} = 0.3989422804014327$$



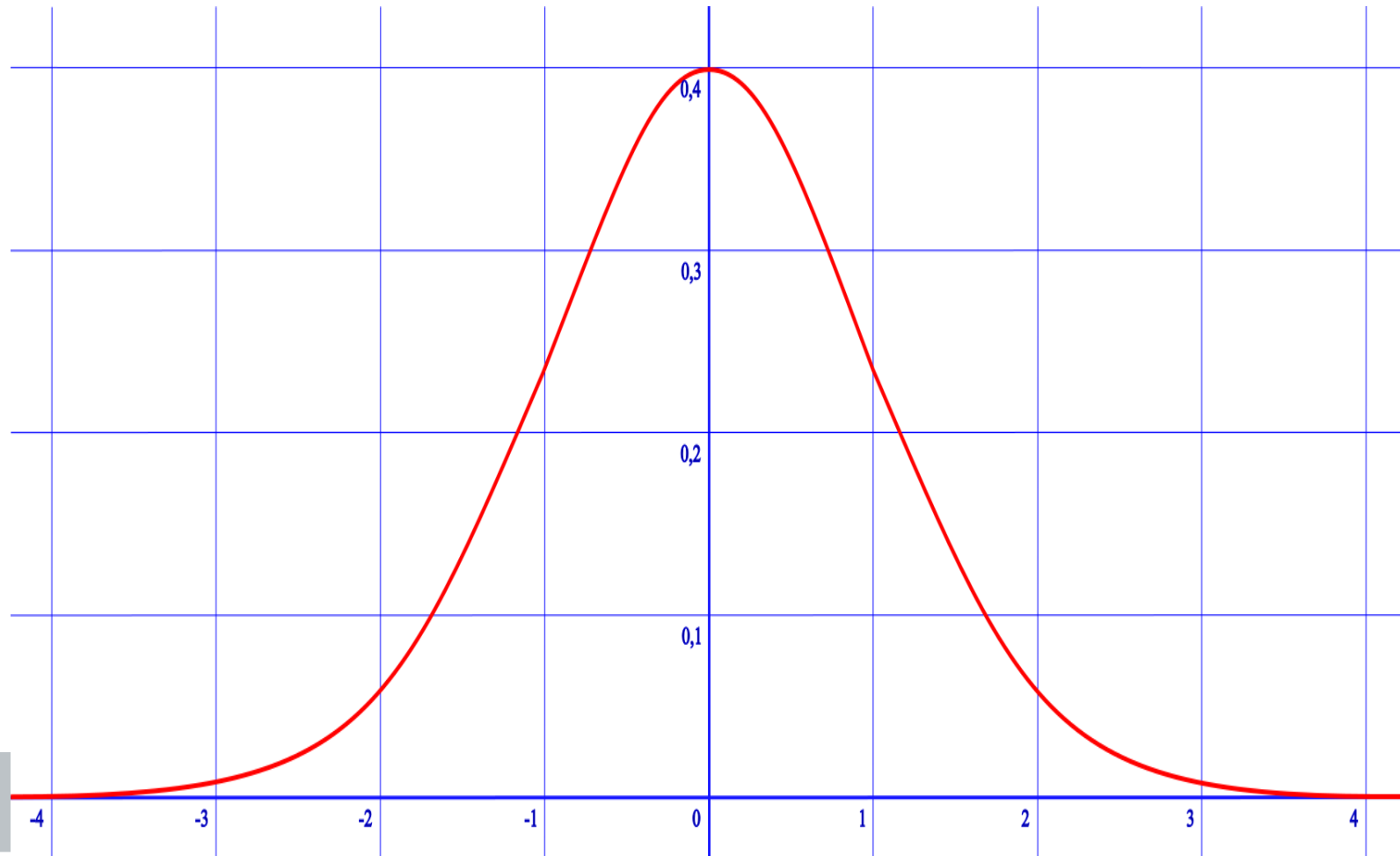
Programación Evolutiva

$$\frac{\left(e^{-\left(0.5 \times ((-2)^2)\right)}\right)}{\sqrt{2 \times \pi}} = 0.0539909665131881$$



Programación Evolutiva

$$\frac{\left(e^{-\left(0.5 \times (1^2)\right)}\right)}{\sqrt{2 \times \pi}} = 0.2419707245191434$$



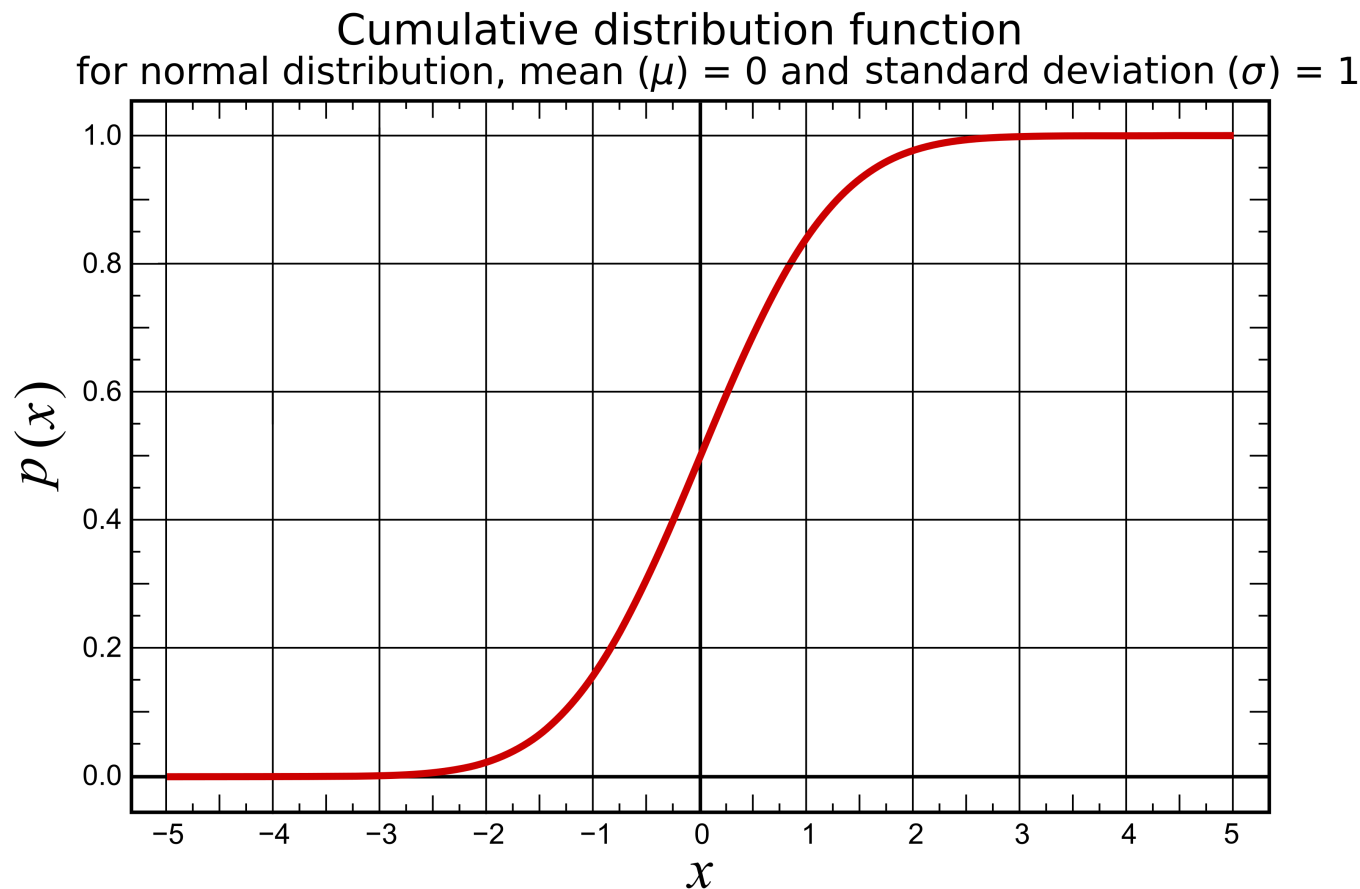
Programación Evolutiva

- Una distribución normal de media cero es implementada de la siguiente manera:

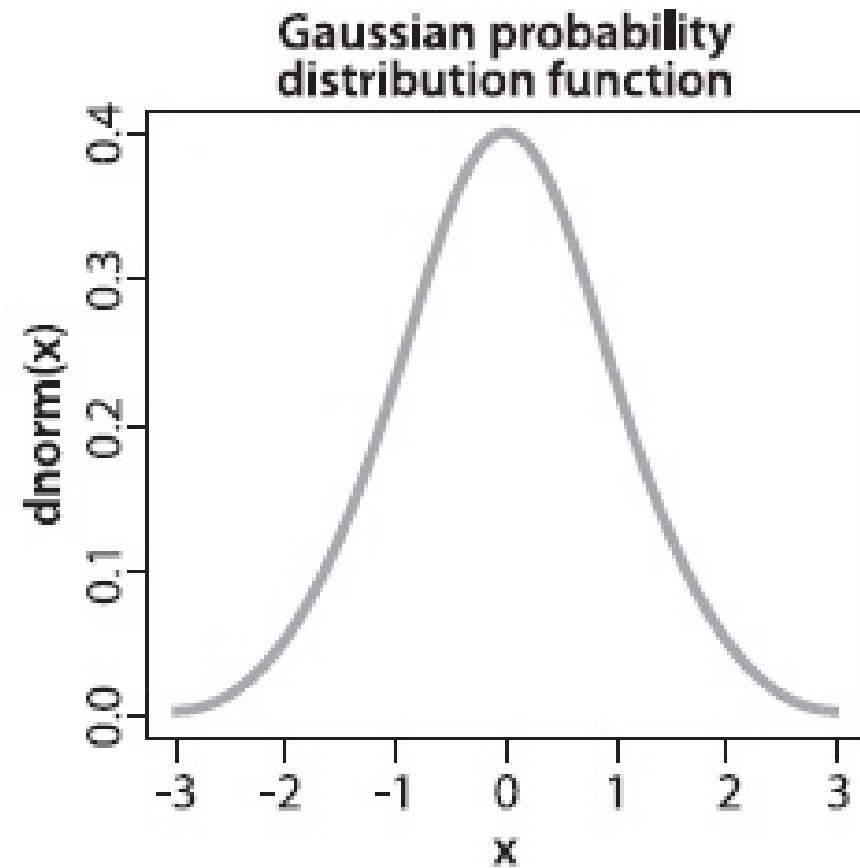
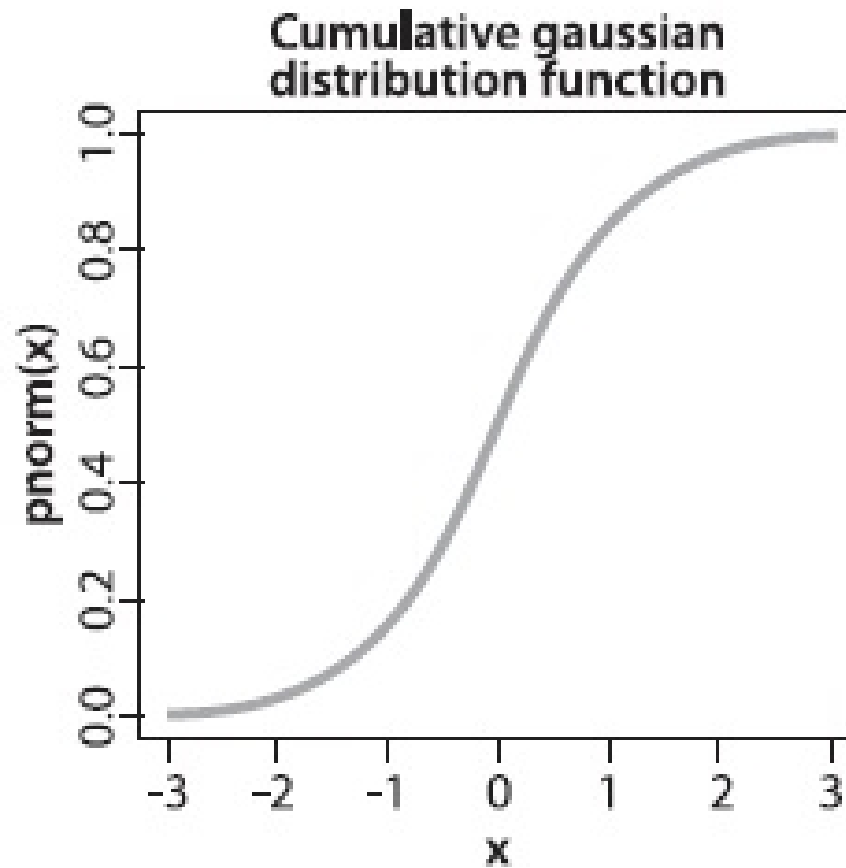
```
public static double normal(double x, double desvio) {  
    double retorno = -0.5 * ((x / desvio) * (x / desvio));  
    retorno = Math.exp(retorno);  
    retorno = retorno / (desvio * Math.sqrt(6.283184));  
    return retorno;  
}
```

Programación Evolutiva

- El área bajo la distribución de probabilidades corresponde a la probabilidad de ocurrencia del valor x . Por lo tanto, como se puede ver en la siguiente figura, el área total de una distribución de probabilidad es igual a 1.



Programación Evolutiva



Programación Evolutiva

- En base a ese concepto, podemos escoger cual será la variación de coordenadas sorteando un valor ϵ aleatorio entre el intervalo $(0,1)$ y determinar el valor de x , para el cual el área bajo la curva hasta x es igual al sorteado, es decir, el número x , para el cual la probabilidad de que un valor sorteado, cualquiera sea menor de que el, sea igual a ϵ .

Programación Evolutiva

- Para determinar esta probabilidad y calcular el valor de la mutación a aplicar, tenemos que calcular el valor de la integral dada por:

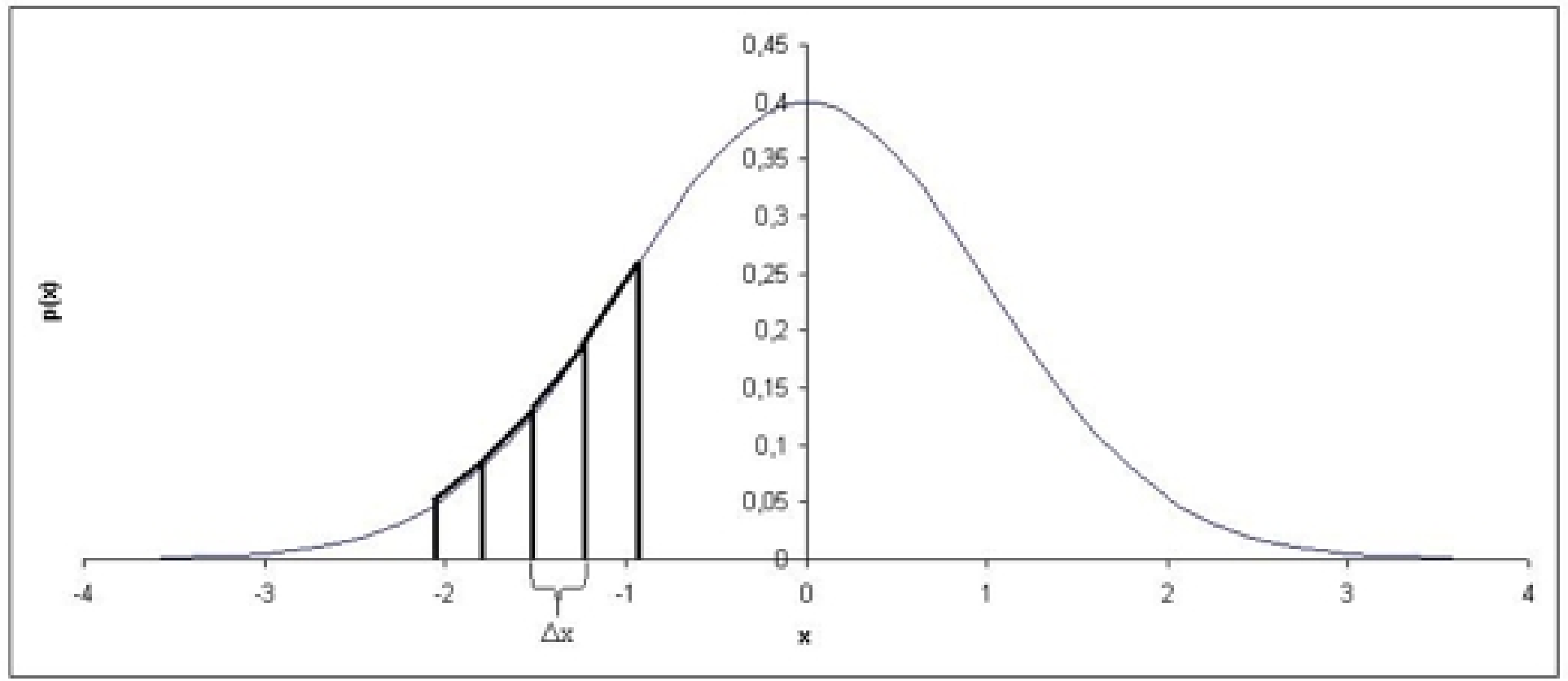
$$\int_{-\infty}^x N(0, \sigma, x) dx$$

- Como sabemos, no existe una forma definitiva para calcular esta integral, para ello usamos técnicas numéricas para implementarla.

Programación Evolutiva

- Usaremos el método de los trapecios repetidos:
- Idea:
 - Aproximar la curva a una serie de trapecios.
 - La base igual a Δx
 - Los dos lados son dados por los valores de la función de los puntos que distan Δx uno del otro.
 - Consiste básicamente en hacer una aproximación lineal por partes de la función que deseamos integrar.
 - La aproximación puede ser tan adecuada como desiemos, basta disminuir el valor de Δx

Programación Evolutiva



Programación Evolutiva

- El código que implementa la integración es el siguiente:

```
public static double integral(double lim_inf, double lim_sup, double desvio, double delta) {  
    double area = 0;  
    double aux_suma, aux = normal(lim_inf, desvio);  
  
    for (double i = lim_inf + delta; i < lim_sup; i += delta) {  
        aux_suma = normal (i, desvio);  
        area += (aux + aux_suma);  
        aux = aux_suma;  
    }  
    area *= (delta/2);  
    return area;  
}
```

Programación Evolutiva

- Al final multiplicamos la suma obtenida por $(\text{delta}/2)$, para obtener el área efectiva.
- Recordemos que el área de un paralelogramo está dada por $\text{base} \cdot (\text{altura1} + \text{altura2})/2$.
- En este caso, los valores de la normal en cada división del intervalo corresponde a las alturas. El valor de la base es igual para todos, por lo que puede ser multiplicado al final.

Programación Evolutiva

- Podemos calcular el valor de x mediante:


```
public static double valor_x(double lim_inf, double lim_sup, double desvio, double delta, double aleatorio) {  
    double area = 0;  
    double aux_suma, aux = normal(lim_inf, desvio);  
  
    for (double i = lim_inf + delta; i < lim_sup; i += delta) {  
        aux_suma = normal (i, desvio);  
        area += (aux + aux_suma);  
        if((area * (delta/2)) > aleatorio){  
            return i;  
        }  
        aux = aux_suma;  
    }  
    return -1*Double.MAX_VALUE;  
}
```

Programación Evolutiva

- Sigue el mismo procedimiento visto anteriormente:

```
Procedure EP{  
    t = 0;  
    Initialize P(t);  
    Evaluate P(t);  
    While (Not Done)  
    {  
        Parents(t) = Select_Parents(P(t));  
        Offspring(t) = Procreate(Parents(t));  
        Evaluate(Offspring(t));  
        P(t+1) = Select_Survivors(P(t), Offspring(t));  
        t = t + 1;  
    }
```

No existe cruzamiento,
solo Mutación



Programación Evolutiva

- Con la diferencia de que la selección de la siguiente población es sobre la población de ascendientes y descendientes.
- Esta modificación también es llamada de Meta-PE o $PE(\mu+\mu)$.

GRACIAS

Dr. Edward Hinojosa Cárdenas
ehinojosa@unsa.edu.pe