Computación Emergente

Prof. Kay Tucci

Facultad de Ciencias

Universidad de Los Andes

Mérida, Venezuela

Justificación

La resolución de problemas complejos a través de nuevos métodos computacionales es fruto de avances en diferentes áreas científicas como la biología, la computación, la física y la estadística, entre otras. A estos métodos se los agrupa bajo el concepto de Computación Emergente y se enfocan hacia la aplicación de técnicas computacionales no convencionales para el estudio de fenómenos y la resolución de problemas que debido a la complejidad que presentan no pueden ser abordados utilizando métodos tradicionales. Actualmente se incluye en la computación emergente a: *autómatas celulares, redes neuronales, algorítmos genéticos, vida artificial y sistemas multiagentes.*

Objetivos Generales

- Adquirir los conceptos básicos de las diferentes herramientas de computación emergente y los casos en los que es conveniente el uso de cada una de ellas.
- Diseñar utilizando las diferentes técnicas de computación emergente modelos y soluciones a problemas.
- Adquirir destreza en el uso de la computación emergente para la investigación de problemas complejos.

CONTENIDO

I. **Introducción.**

Ejemplos de sistemas en los que se requiere el uso de la **CE**.
Complejidad y **CE**

II. **Autómatas Celulares. (CAs)**

Conceptos básicos de los **CA**s.
Reglas de transición, Vecindad, Condiciones de Borde.
Los **CA**s como herramientas de Modelado.
Ejemplos: Crecimiento de superficies, Dinámicas de opinion, Dinámica molecular, Pilas de arena, Tráfico, Movimiento de cuerpos sólidos, etc.

III. **Algorítmos Genéticos. (GAs)**

Conceptos básicos de **GA**s. Función Objetivo y Operadores evolutivos.
Ejemplos: Optimización de recursos, agente viajero, control

adaptativo.

IV. **Redes Neuronales.**

Conceptos básicos de redes neuronales.
Aprendizaje y algoritmos de entrenamiento.
Tipo de redes neuronales: perceptrónicas multicapas, dinámicas y auto-organizables.
Ejemplos: Problemas de clasificación, asociación y aproximación.

V. **Sistemas Multiagentes.**

Conceptos básicos de sistemas multiagentes.
Tipos de agentes. Comunicación entre agentes.
Lógica de primer orden y motores de inferencia.
Diseño de agentes: Creencias, metas, base de conocimiento e interacción con el entorno.
Ejemplos: Cooperación-confrontación, dilema del prisionero.

VI. **Tópicos Recientes.**

Computación cuántica.

Compotación basada en caos.

Bioinformática y vida artificial.

Sistemas Complejos

Son sistemas compuestos por ***múltiples elementos que interactúan entre sí y cuyo comportamiento global es emergente***, es decir que no pueden explicarse completamente a partir del conocimiento de las características de los elementos.

Un mismo sistema puede ser catalogado de complejo y simple a la vez, dependiendo del aspecto que se considere.

El tratamiento y comprensión de los fenómenos que se presentan en los sistemas complejos utilizando un enfoque reduccionista[1] es, en general, inadecuado.

[1] El reduccionismo se resume en la frase "*Divide cada problema en tantas partes como sea posible y necesario para resolverl o*", expresada por el filósofo, médico y matemático francés René Descartes en el siglo XVII.

Sistemas Complejos

No existe en la literatura actual un concepto único de Sistemas Complejos, adoptaremos el punto de vista de sistemas complejos como un conjunto de elementos, interactuantes entre sí en varios niveles jerárquicos, de tal forma que el sistema en su globalidad exhiba comportamientos y propiedades emergentes, las cuales no se pueden inferir del análisis de los elementos constituyentes.

Sistemas con estas características han sido denominados anteriormente como *Estructuras Disipativas* por I. Prigogine, *Sistemas Sinergéticos* por H. Haken y *Sistemas Auto-organizados* por P. Bak.

Modelado de Sistemas Complejos

Tradicionalmente, el modelado de sistemas ha sido utilizado con el fin de que, mediante la simulación de dichos modelos se logre imitar la realidad.

El modelo desde el punto de vista clásico es una representación válida de un sistema real que utiliza un conjunto de instrucciones, reglas, ecuaciones y restricciones para generar el comportamiento Entrada/Salida del sistema real.

Al modelar sistemas complejos con técnicas tradicionales o reduccionistas a lo sumo pueden garantizar en algunos casos la validez replicativa, ya que el modelador no puede establecer a priori como afectan al comportamiento del modelo las simplificaciones que se introducen al modelar.

Modelo

Es una representación válida de un sistema real que utiliza un conjunto de instrucciones, reglas, ecuaciones y restricciones para generar el comportamiento Entrada/Salida del sistema real.[1]

El proceso de modelado *siempre* implica perdida de información.

Existen tres niveles de validación del modelo:

1. **Validez Replicativa**: Reproduce resultados del sistema real
2. **Validez Predictiva**: Predice comportamientos
3. **Validez Estructural**: Preserva las estructuras del sistema real

El tercer nivel es el más fuerte y pretende imitar paso a paso y componente a componente la manera en que el sistema real realiza sus transiciones de estado.

[1] B. Zeigler, et al. Theory of Modelling and Simulation. Academic Press, 2000

Modelado de Sistemas Complejos

El modelar Sistemas Complejos como pequeños cambios en los modelos que los representan pueden acarrear grandes cambios en los resultados de la simulación.

Esta propiedad I. Tsuda la ha denominado **Inestabilidad Descriptiva**.

K. Kaneko y I. Tsuda proponen la necesidad de crear un nuevo método de modelado en el que se crea un mundo artificial o virtual, el cual se construye de tal forma que presente el fenómeno de interés y mediante la simulación se pueda obtener una descripción cualitativa del fenómeno. A esta forma de modelar se le conoce con el nombre de **Modelado Constructivista**.

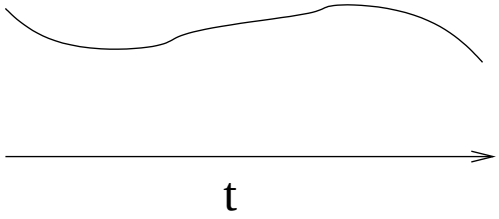
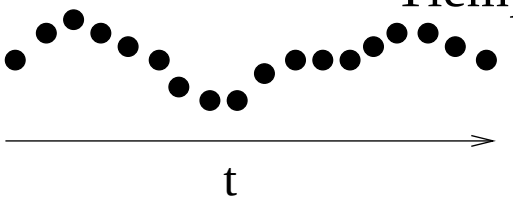
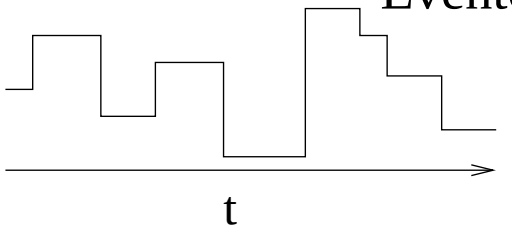
Enfoque Constructivista del Modelado

Se construye un *Modelo* para observar y estudiar un *Fenómeno*

- No se pretende imitar a un sistema real por lo que se incluyen en el modelo los componentes necesarios para el estudio del fenómeno
- La simplicidad del *Modelo* no implica perdida de información
- La validez del *Modelo* es estructural porque el sistema real y el fenómeno son una unidad.

La generalidad de los modelos así creados permite estudios cualitativos de comportamientos universales presentes en los Sistemas Complejos.

Formalismos para el Estudio de Sistemas Dinámicos

Sistema	Modelo	Simulador
Ecuaciones Diferenciales 	dq/dt = f(q)	Integrador Numérico
Tiempo Discreto 	x(t+1) = f(x)	Algoritmo Recursivo
Eventos Discretos 	hora y estado del evento	Procesador de Eventos

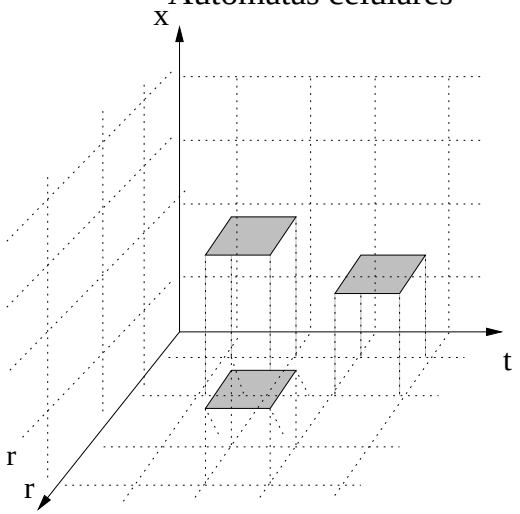
Sistemas Dinámicos Espacialmente Distribuidos

Una forma de modelar Sistemas Complejos es mediante la construcción de Sistemas Dinámicos Espacialmente Distribuidos, los cuales se pueden clasificar en:

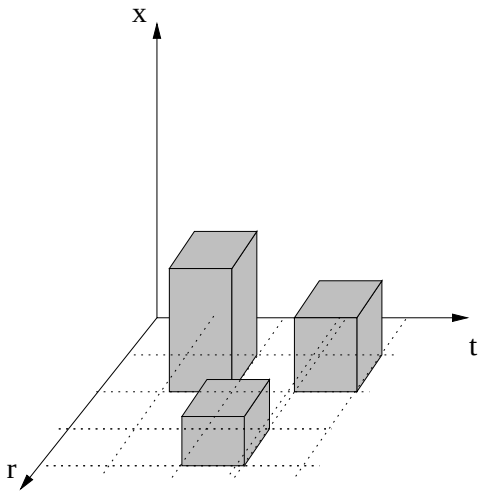
Modelo	Tiempo	Espacio	Estado
Autómatas Celulares	D	D	D
Redes de Mapas Acoplados	D	D	C
Mapas acoplados en un Contínuo	D	C	C
Ecuaciones Diferenciales Ordinarias	C	D	C
Ecuaciones Diferenciales Parciales	C	C	C

Sistemas Dinámicos Espacialmente Distribuidos

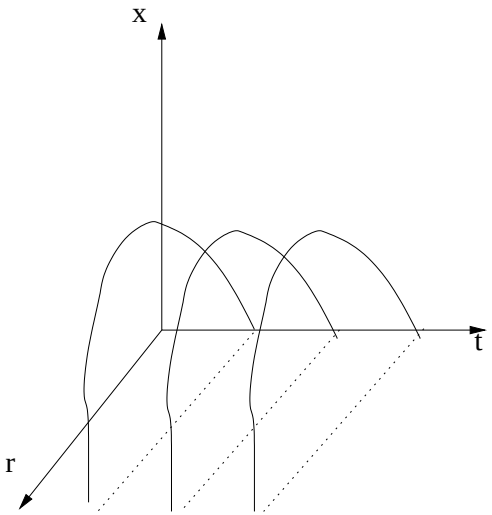
Autómatas celulares



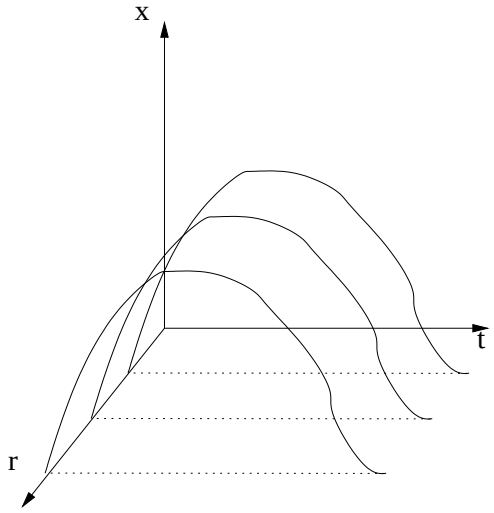
Red de mapas acoplados



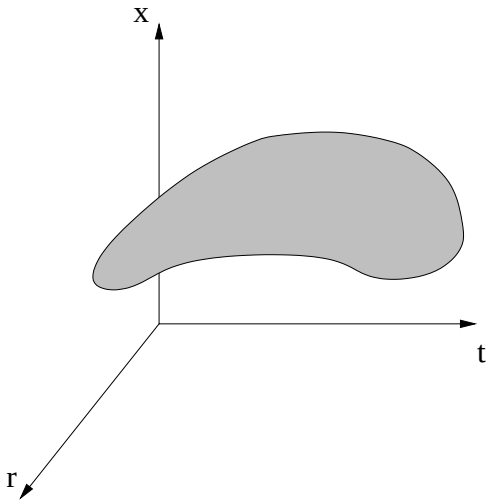
Mapas acoplados en un continuo



Ecuaciones diferenciales ordinarias



Ecuaciones diferenciales parciales



Introducción a los CAs

Los **CA**s y las técnicas de modelado relacionadas con ellos conforman un método poderoso para estudiar el comportamiento de *Sistemas Complejos* porque permiten:

- **Describir** Expresar en forma clara y sintética las características *relevantes* de algunos Sistemas Complejos
- **Entender** Comprender los orígenes de los comportamientos colectivos *emergentes*
- **Simular** Realizar experimentos en al computador con sistemas de tamaño y duración *suficientes*

¿Qué son y cuándo nacieron los CA?

Son idealizaciones de sistemas espacialmente distribuidos en las que el espacio y el tiempo son discretos y las cantidades físicas toman un conjunto de valores finitos.

La idéa ...

En los años 40s John Von Neumann buscaba imitar el comportamiento del cerebro para poder construir un computador capaz de resolver problemas complejos, este computador debía estar compuesto por un conjunto de elementos simples que interactuaban entre sí. Ulam le sugiere a Von Neumann discretizar el tiempo, el espacio y los estados del sistema.

Entonces podemos decir que en los CAs ...

Los estados del sistema evolucionan como un conjunto de autómatas ubicados sobre una retícula similar al espacio celular biológico.

El primer CA lo diseñó Von Neumann:

- Retícula cuadrada.
- más de 1000 celdas o células
- 29 estados posibles
- Cada celda interactuaba con sus 4 primeros vecinos

No se pudo implementar completamente, pero ...

Ruptura de un paradigma

Se encontró que una estructura discreta de celdas podía contener dentro de sí la forma de generar nuevos individuos identicos.

Viejo paradigma Una màquina sòlo puede crear objetos de menor complejidad que la de la misma máquina.

Nuevo Paradigma: Con un máquina (CA) se puede crear una nueva máquina (CA) de idéntica complejidad y capacidad.

Muy breve evolución histórica

- Años 50 se usaron para procesar imágenes.
- 1970 John Conway diseño el *Game of Life* encontró reglas simples que generaban comportamiento complejo. Surgían espontaneamente estructuras (*gliders*)
- Años 80 Wolfram observa el comportamiento y estudia sistemáticamente como sistemas dinámicos discretos una familia de CA. unidimensionales (Reglas de Wolfram).
- Años 80 Toffoli y Margolus construyeron la primera CAM ,*Cellular Automata Machine*, de propósito general CAM-6.

Otra definición

" . . . los CAs son modelos sintéticos del universo en los que las leyes físicas son expresadas en términos de reglas locales y simples sobre una estructura espaciotemporal discreta . . . "

T. Toffoli 1987.

¿Por qué modelar con CAs ?

" . . . los CAs ofrecen un marco conceptual, como también una eficiente herramienta numérica, que conserva los aspectos microscópicos de las leyes físicas como son:

- *Simultaneidad de movimiento*
- *Interacciones locales*
- *Reversibilidad del tiempo"*

J.C. Maxwell

" . . . Modelar con CAs permite capturar las características escenciales de un fenómeno y trasladarlas a una estructura numericamente eficiente"

B. Chopard y M. Droz

Extensiones de modelos físicos con CAs

Modelos de *Gas Lattice* o HPP[1] fue adaptado a CAs en 1980 por Maxwell, quien propuso discretizar la velocidad de las partículas.

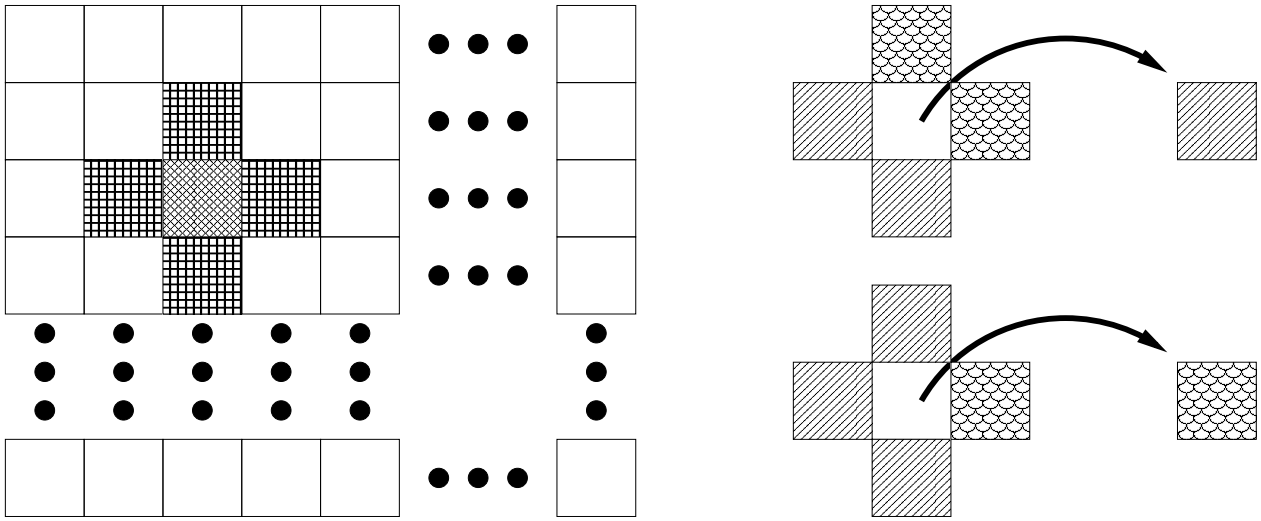
Modelos de FHP[2] podían sustituir a los túneles de viento dentro de ciertos límites, reproduciendo resultados obtenidos con modelos hidrodinámicos basados en las equaciones de Navier-Stockes.

- Flujos en medios porosos
- Erosión
- Reacción difusión
- Crecimiento por agregación y nucleación

[1] Desarrollado por Hardy, Pomeau y Pazzis en los años 70

[2] Desarrollado por Frisch, Hasslacher y Pomeau y simultaneamente por Wolfram en 1986

Ejemplo: Regla de Paridad



$$x_{t+1}(i,j) = x_t(i+1,j) \oplus x_t(i-1,j) \oplus x_t(i,j+1) \oplus x_t(i,j-1)$$

donde, x es el estado, t el tiempo e i, j la posición de la celda, con:

$$x = \begin{cases} 0 & y \quad 0 \oplus 0 = 1 \oplus 1 = 0 \\ 1 & y \quad 0 \oplus 1 = 1 \oplus 0 = 1 \end{cases}$$

Definición

Un CA requiere:

- Una estructura de celdas en un espacio de d dimensiones
- Un conjunto de variables discretas

$$\mathbf{x}_t(\mathbf{r}) = \{x_t^1(\mathbf{r}), x_t^2(\mathbf{r}), \ldots, x_t^n(\mathbf{r})\}$$

que definen el estado de la celda ubicada en la posición $\mathbf{r}$ el cual evoluciona en intervalos discretos de tiempo t

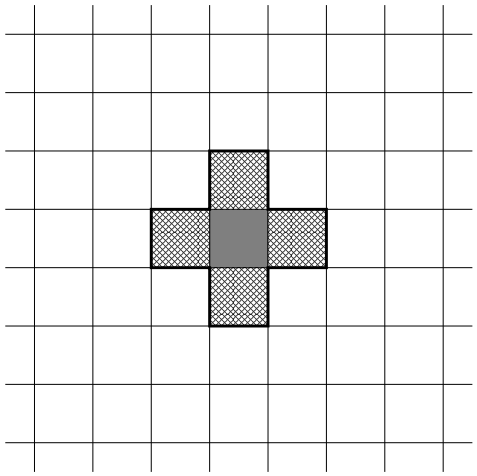
- Un conjunto de reglas $\mathbf{R} = \{R_1, R_2, \ldots, R_m\}$ las cuales especifican como cada uno de los estados de cada celda evoluciona:

$$x_{t+1}^i(\mathbf{r}) = R_j(\mathbf{x}_t(\mathbf{r}), \mathbf{x}_t(\mathbf{n}_1^\mathbf{r}), \ldots, \mathbf{x}_t(\mathbf{n}_q^\mathbf{r}))$$

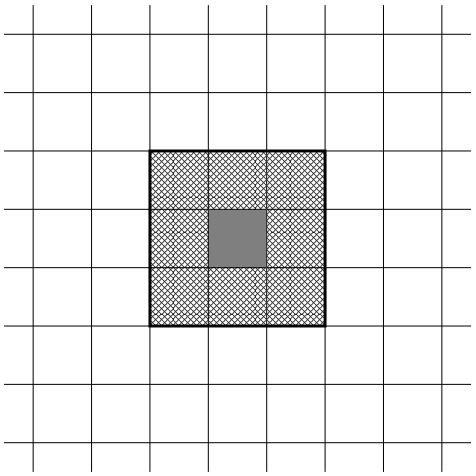
donde, $\{\mathbf{n}_k^\mathbf{r} \quad \forall \quad k = 1, \ldots, q\}$ representa el conjunto de las posiciones de los q vecinos de la celda ubicada en $\mathbf{r}$

Vecindad

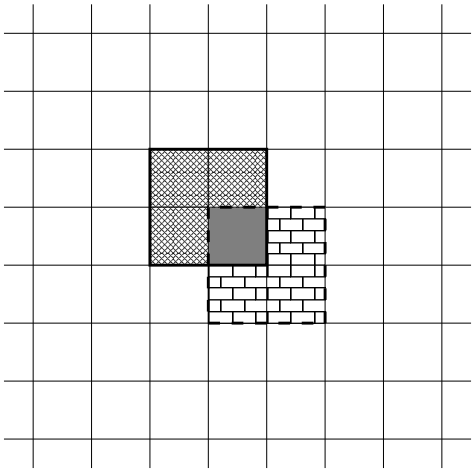
Los reglas dinámicas de los CAs son por definición *locales*, es decir que el estado de cada celda depende solamente del estado de las celdas de su entorno. A este entorno se le conoce como **Vecindad**.



Von Neumann



Moore

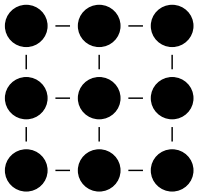


Margolus

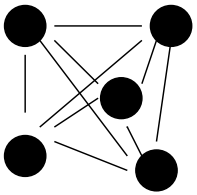
Otros tipos de Vecindad

Es posible definir CAs con otros tipos de vecindades contenidas en espacios euclidianos de dimensión $d \neq 2$:

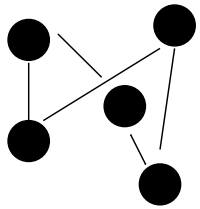
Euclidiano



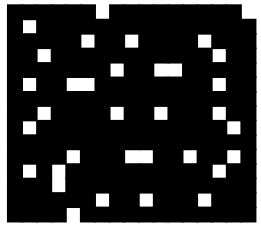
Global



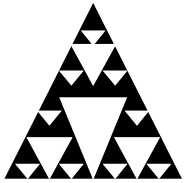
Aleatorio

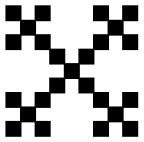


Porosos

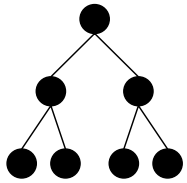


Fractales multi y monoconexos

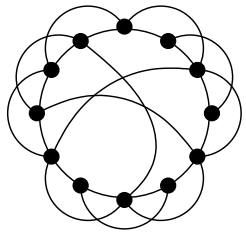




Arboles

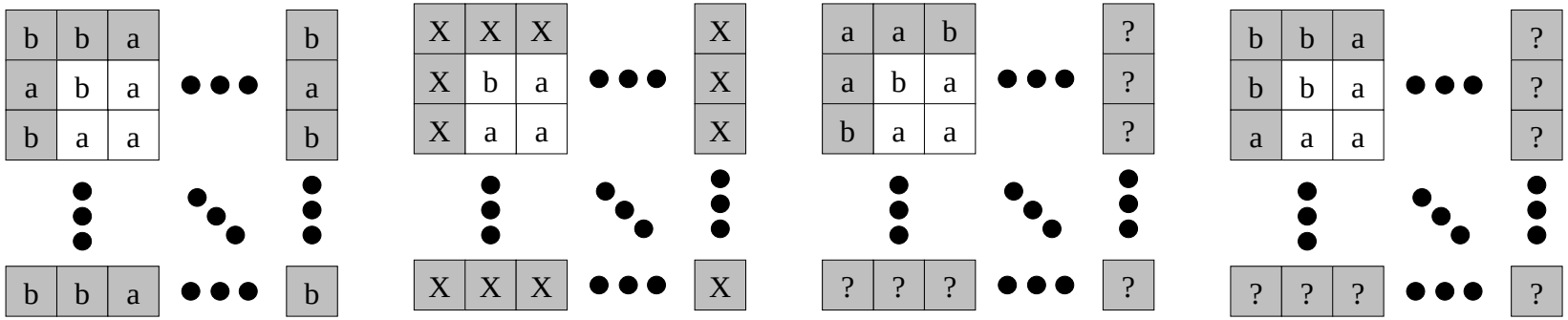


Small World



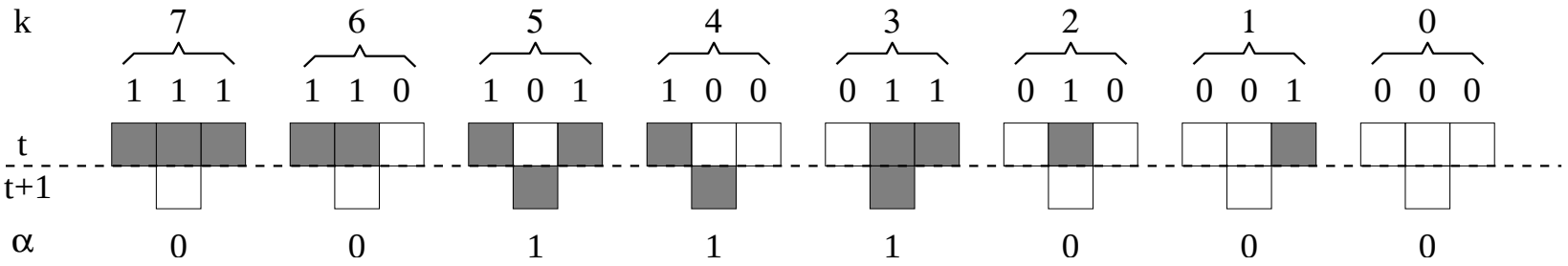
Condiciones de Contorno

Cuando utilizamos **CAs** para modelar algún sistema el espacio sobre el cual construimos el modelo es finito, por esto al diseñar **CAs** es importante considerar las *condiciones de borde o de contorno* del sistema.



Reglas de Wolfram (1983)

El CA es unidimensional y con condiciones de borde periódicas. El estado $x_t(i)$ es binario y depende de $\{x_t(i-1), x_t(i), x_t(i+1)\}$



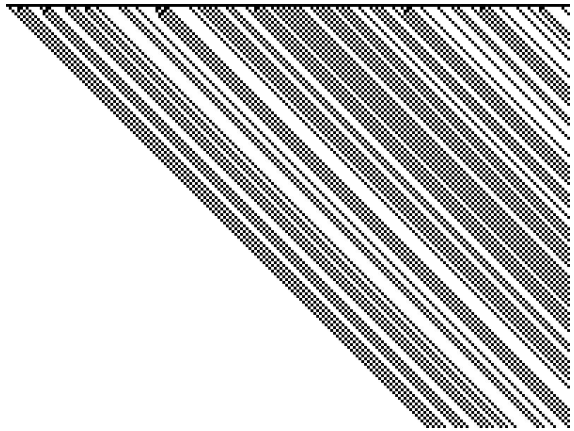
$$N_w = \sum_{k=0}^{7} 2^k \alpha_k$$

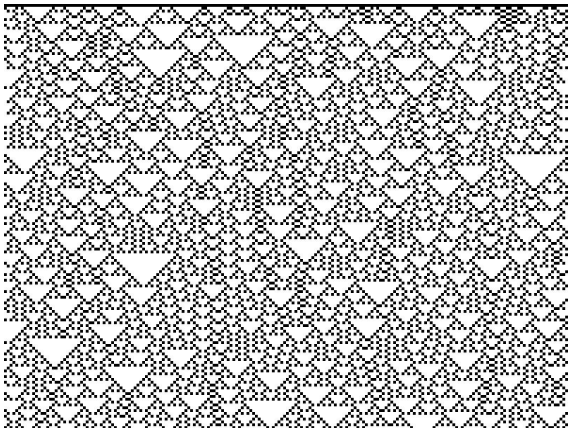
http://www.ciens.ula.ve/~kay/Academ/CompEmergDemos

Cuatro tipos de Comportamiento

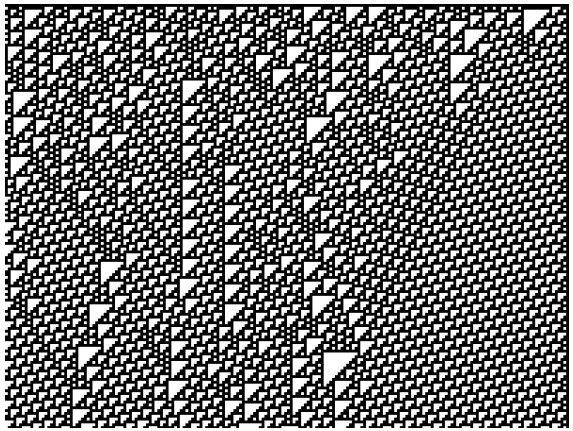
Punto Fijo

Ciclo Límite





Atractor Extraño

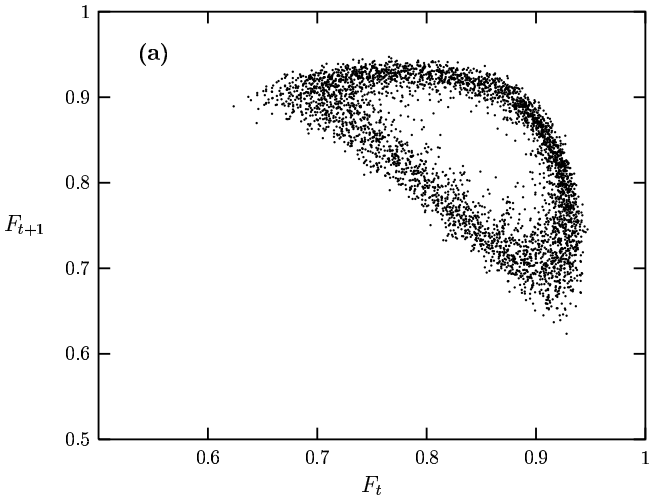


Estructuras Complejas

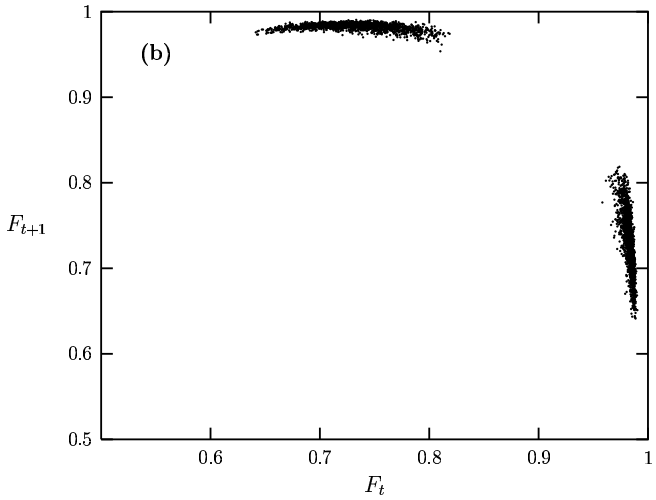
Comportamiento Colectivo no Trivial

En modelos construidos con CAs también se ha observado el surgimiento de *Comportamiento Colectivo No Trivial* que se refiere a el comportamiento *inesperado* de cantidades macroscópicas como por ejemplo el promedio.

Cuasiperiódico



Período 2



Fracción de elementos turbulentos en redes *smallworld*

Prob. Reconexión: $p = 0{,}80$; Acoplamiento: a) $\epsilon = 0{,}67$; b) $\epsilon = 0{,}85$

Comportamiento Colectivo no Trivial

La aparición de Comportamiento Colectivo No Trivial en Sistemas Complejos depende de los valores de los parámetros del sistema. Con esos valores de parámetros éste comportamiento es estable y robusto.

El Comportamiento Colectivo No Trivial aparece tanto en sistemas determinísticos como en sistemas probabilísticos y requiere de sincronismo en la actualización de los estados y comunicación instantánea de los mismos[1]

[1] Chaté y Maneville 1992

Modelo de Propagación de Incendios

Unas reglas probabilísticas sencillas para modelar la propagación de incendios forestales con un CA de un solo estado que puede tomar tres valores

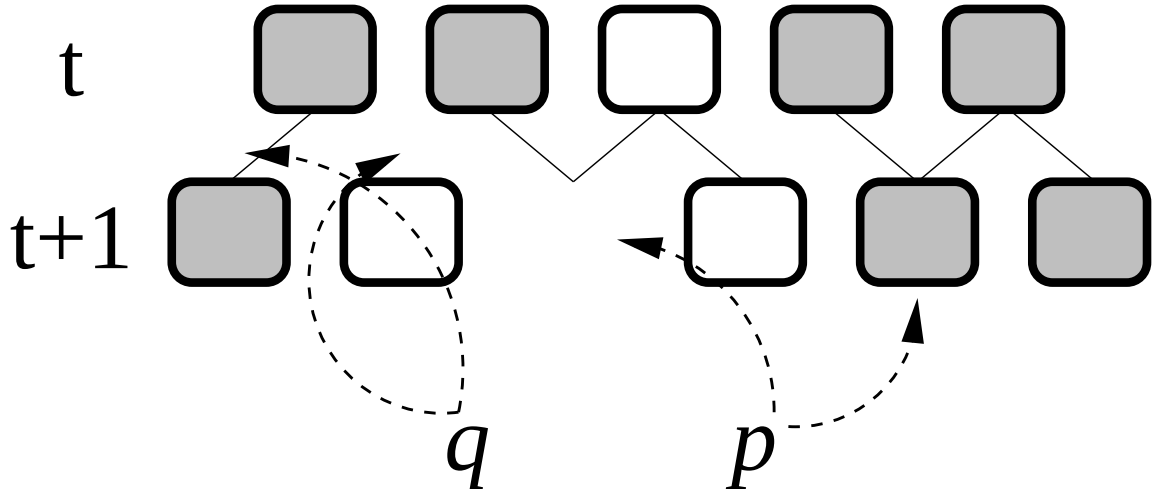
$$x_t(i,j) = \begin{cases} 0 & \text{, No hay árboles ni fuego;} \\ 1 & \text{, Árboles (combustible);} \\ 10 & \text{, Fuego} \end{cases}$$

Siendo la regla:

$$x_{t+1}(i,j) = \begin{cases} 0 & \text{, si } x_t(i,j) = 10; \\ 10 & \text{, si } x_t(i,j) = 1 \text{ y } \sum_{vecinos} x_t(\mathbf{v}) \geq 10; \\ 1 & \text{, con probabilidad } p \text{ si } x_t(i,j) = 0 \ ; \\ 10 & \text{, con probabilidad } q \text{ si } x_t(i,j) = 1; \end{cases}$$

Modelo de Percolación

Un modelo sencillo de percolación en medio poroso con probabilidad p de que exista un poro y probabilidad q



Se encuentran valores críticos de $p = p_c$ y $q = q_c$ para los cuales el sistema percola completamente. El comportamiento del las cantidades físicas cuando el sistema se encuentra cerca de p_c y q_c presentan comportamientos que siguen leyes de potencia con exponentes críticos universales.

Separación de Fases o Dominios

El modelo propuesto por G. Vichniac en 1984:

$$x_t(i,j) = \begin{cases} 0 & \text{, si } \sum_{vecinos} x_t(\mathbf{v}) \in \{0,1,2,3,5\}; \\ 1 & \text{, si } \sum_{vecinos} x_t(\mathbf{v}) \in \{4,6,7,8,9\}; \end{cases}$$

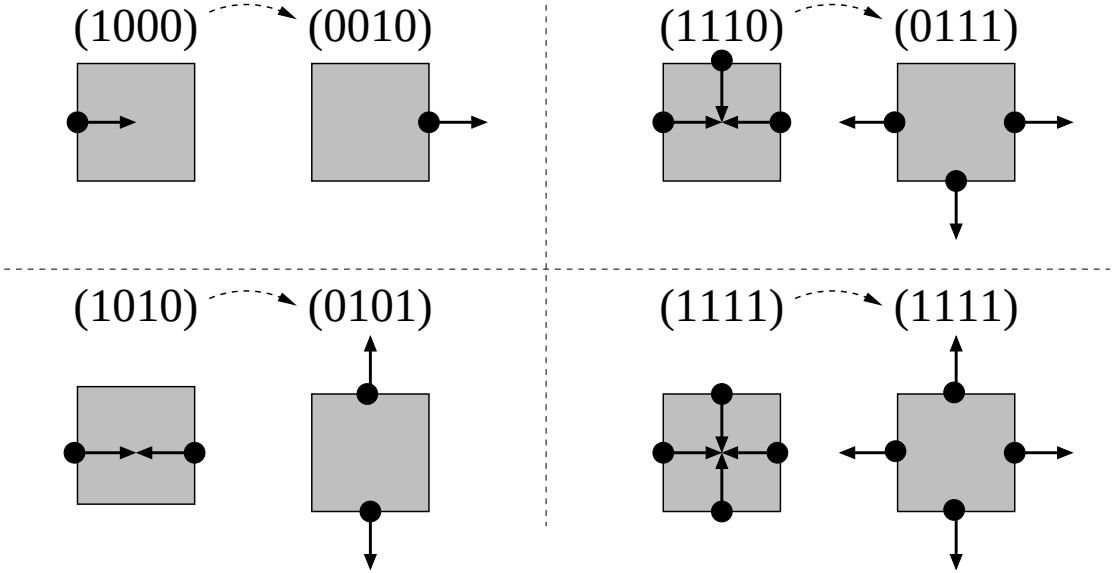
permite modelar algunos mecanísmos de separación de fases. En el modelo se observa que la velocidad normal de la interfaz es proporcional a la curvatura, característico de varios fenómenos.

Esto ocurre a pesar de que la dinámica de cada elemento no toma en cuenta la formación de dominios e interfaces en el sistema.

Dinámica de Partículas (Modelo HPP)

El modelo HPP, propuesto por Hardy, Pomeau y Pazzis en los años 70, consiste en una estructura sobre la cual se mueven partículas en dos pasos: *colisión* y *desplazamiento* teniendo en cuenta

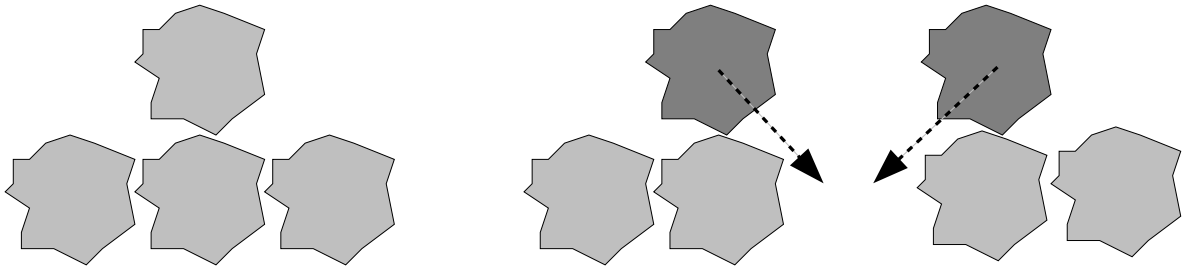
- El principio de exclusión
- Conservación de momento



El modelo HPP es reversible por ser numéricamente exacto.

Material Granulado

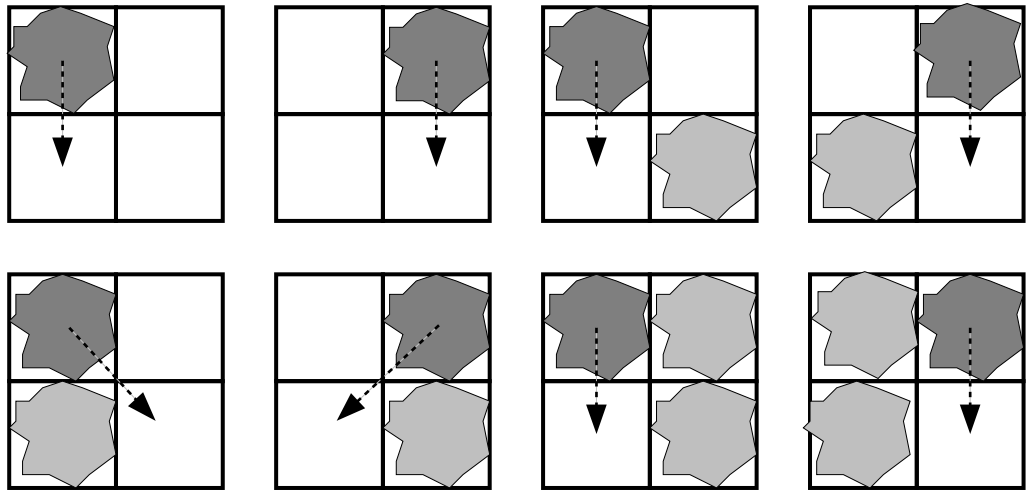
La cinética de material granulado se basa en que un grano se conserva sobre otro y no cae si la estructura que lo sostiene es estable.



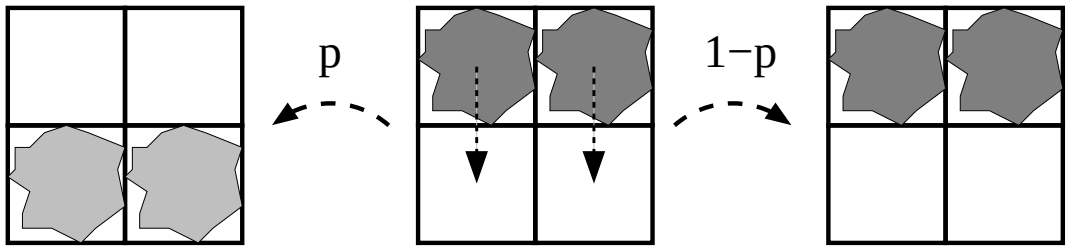
Para resolver el conflicto que se presenta por el principio de exclusión se puede ampliando el vecindario de cada celda para preveer esta situación o ...

Modelos con Material Granulado

Definiendo el vecindario tipo Margolus:



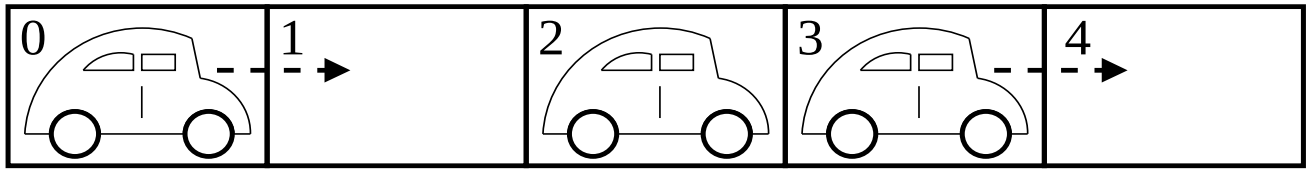
Donde inclusive se puede definir un bloqueo por fricción



Modelos de Tráfico

El modelo se basa en el hecho un carro solamente puede avanzar si al sitio donde va está desocupado. Si la celda a la que el carro quiere moverse está ocupada, el movimiento no se dará, a pesar de que quien ocupa la celda destino se moverá en la iteración actual.

$$x_{t+1}(i) = x_t(i-1)(1 - x_t(i)) + x_t(i+1)x_t(i)$$



Se pueden agregar al modelo paradas aleatorias (carga y descarga de pasajeros) u obligadas (semáforos) semáforos

Introducción a las NNs

Las NNs han despertado un interez creciente en los últimos años, y se están aplicando con éxito en muchos problemas, en las áreas diversas como: finanzas, medicina, ingeniería, geología y física, entre otros. De hecho, las NNs pueden usarse en problemas de predicción, clasificación y control. Su éxito puede atribuirse principalmente a dos factores:

- **Potencia**: Las NNs son técnicas no lineales sofisticadas capaces de modelar funciones extremadamente complejas de alta dimensionalidad.
- **Fáciles de usar**: Las NNs aprenden con el ejemplo. El usuario recopila datos representativos, y luego utilizando un *algoritmo de entrenamiento* ipara que la NN aprenda a partir de los datos. Es decir que el usuario solamente requiere conocer el comportamiento deseado de la NN pero no el cómo lograrlo.

La neurona natural

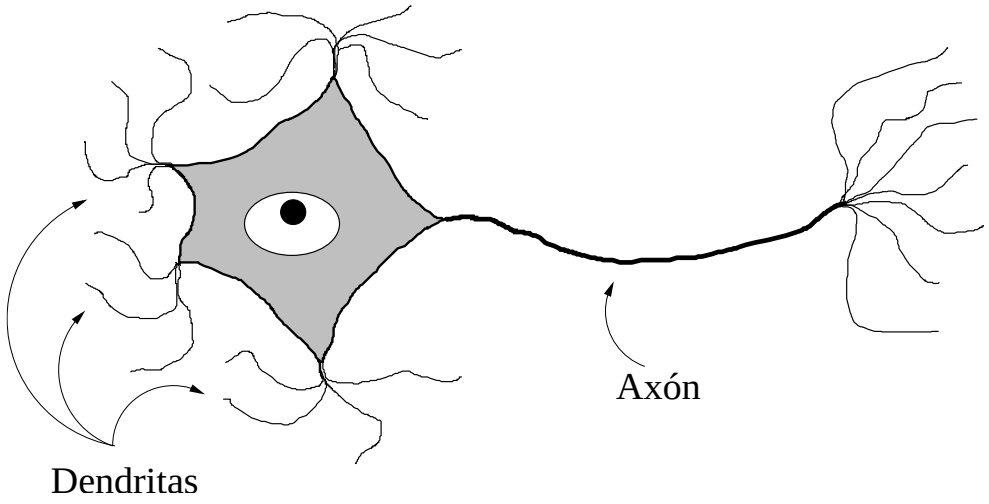
La neurona es una célula especializada que puede propagar una señal electroquímica. La neurona tiene una estructura ramificada de la entrada llamada dendritas, un cuerpo de la célula, y una estructura ramificada de la salida, el axión. Los axones de una célula se conectan con las dendritas de otras neuronas a través de sinapsis.

Cuando se activa una neurona, dispara una señal electroquímica a lo largo del axón. Esta señal cruza las sinapsis hacia otras neuronas, las cuales podrían activarse también.

Una neurona se activa solamente si la señal total recibida en el cuerpo de la célula a través de las dendritas excede cierto nivel, el *umbral de disparo*.

¿Qué son las NNs?

Una NN es una red de muchos procesadores simples ("*neuronas*"), cada uno de los cuales puede tener una cantidad pequeña de memoria local. Las unidades están conectadas por canales de comunicaciones ("conexiones") que transmiten datos codificados. Las unidades funcionan solamente con datos locales; las entradas se reciben a través de algunas conexiones ("*dendritas*") y la salida es transmitida por otras conexiones ("*axones*").



El Cerebro, una red neuronal natural

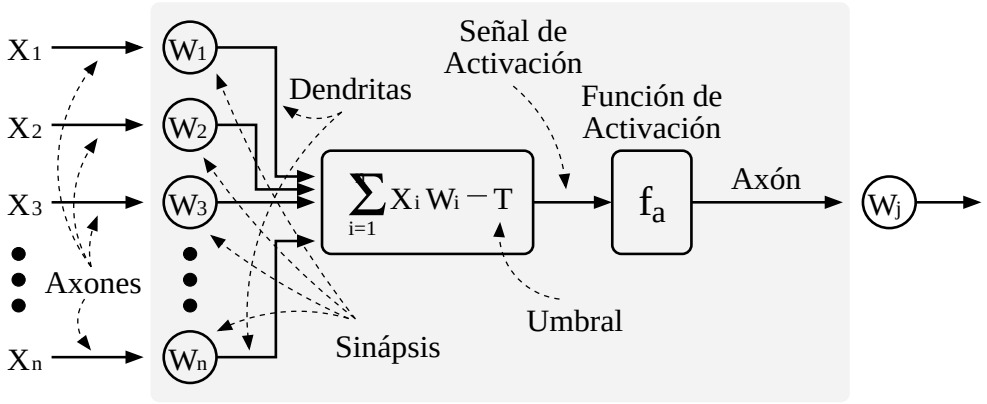
El cerebro se compone principalmente de un número muy grande (10^{10} aprox.) de neuronas, que están altamente interconectadas entre sí, en promedio más de 1000 sinapsis por neurona.

La sinapsis es una separación entre el axón de una neurona y la dendrita de otra. Este espacio está lleno de sustancias neurotransmisoras las cuales permiten transmitir la señal de una neurona a otra. La intensidad de la señal recibida por una neurona depende de la eficacia de las sinapsis.

Donald Hebb (1949) postuló que aprender consistía principalmente en alterar la "fuerza" o eficiencia de las conexiones sinápticas.

La neurona artificial

En 1943 McCulloch y Pitts introdujeron un modelo de neuronas que podía implementarse con circuitos. Una neurona artificial:



- Recibe varias entradas ponderadas (*pesos*) por sus dendritas. Los *pesos* corresponde a la eficacia sináptica biológica.
- La suma ponderada de las entradas se resta al umbral y si el resultado es positivo (señal de activación) la neurona se dispara.
- La señal de la activación se pasa a la función de la activación o transferencia para producir la señal de salida de la neurona.

Algunos usos de las NNs ...

Las NNs son especialmente útiles en los problemas de clasificación aproximación y transformación; que sean tolerantes de una cierta imprecisión y de los que se disponga de porciones de datos (pares entrada/salida) para el entrenamiento; especialmente si las soluciones clásicas no son aplicables con facilidad.

Es importante entender que no hay métodos para entrenar a NNs que pueda crear mágicamente la información que no esté en los datos del entrenamiento.

Actualmente, simular con NNs el conocimiento y la emoción humana es ciencia ficción. Las NNs artificial puede ser útiles para modelar algunos aspectos del conocimiento.

Clasificación y Regresión

Los problemas de predicción con NNs se pueden dividir en:

- **Clasificación**: Determinar a cuáles de un número de clases discretas pertenece un caso dado de la entrada. Los ejemplos incluyen la asignacin del crdito, deteccin del cáncer, reconocimiento de la firma, entre otros
- **Regresión**: Predecir el valor de la variable generalmente continua como por ejemplo el precio futuro, el consumo de combustible, los beneficios de una inversión.

En la mayoría de los casos, las NNs tienen una sola variable de salida que puede estar compuesta por múltiples unidades de salida.

Clasificación de las NNs

Según el algoritmo de aprendizaje:

- **Supervisadas**: El comportamiento E/S deseado es conocido y la NN se entrena, ajustando los *pesos* para que su salida coincida con los valores deseados. Una vez entrenada la NN "sabrá" como comportarse.
  - **Auto-asociativo**: El valor de salida es igual que el de la entrada.
  - **Hetero-asociativo**: El valor de la salida es diferente que el de la entrada.
- **No supervisadas**: A la NN no se le proporciona ningún conjunto E/S. Generalmente este tipo de NNs se utilizan para compresión de datos.

Clasificación de las NNs

Según su topología:

- **Feedforward**: Las conexiones entre las *neuronas* no forman ciclos. Generalmente responden rpidamente y la mayora de ellas se pueden entrenar usando mtodos convencionales
- **Feedback**: Hay ciclos en las conexiones. Cada vez que se le da una entrada la NN debe iterar por un tiempo potencialmente largo antes de que produzca una respuesta. Generalmente es ms difcil de entrenar que las *feedforward* NNs.

Las Feedforward se utilizan cuando el conjunto de posibles valores de entrada y salida están acotados, en cambio las Feedback NNs se utilizan cuando estos conjuntos no están acotados, por ejemplo en procesamiento de señales y series temporales.

Clasificación de las NNs

Según el tipo de datos que manejan:

- **Cualitativo**: Las variables solamente toman un número finito de valores de los cuales generalmente varios pertenecen a una misma categoría. Este tipo de NNs se utilizan en problemas de *clasificación*.
- **Cuantitativo**: Las variables son numéricas y representan medidas de una cualidad. Las variables deben tomar valores tales conserven alguna relación aritmética con las cualidades que están representando.

Las primeras decisiones que se deben tomar son: qué variables se van a utilizar; cuántos y cuáles casos se tendrán en cuenta.

Entrada / Salida en NNs Cualitativas

Las NNs con variables cualitativas pueden ser de dos estados `(Encendido, Apagado)` o de múltiples estados `(Rojo, Verde, Azul)`. La numeración puede ser:

- Ordinal { `Rojo=1, Verde=2, Azul=3`}

  Puede confundir la aritmética de la NNs. `Verde` no es el promedio de `Rojo` y `Azul`.

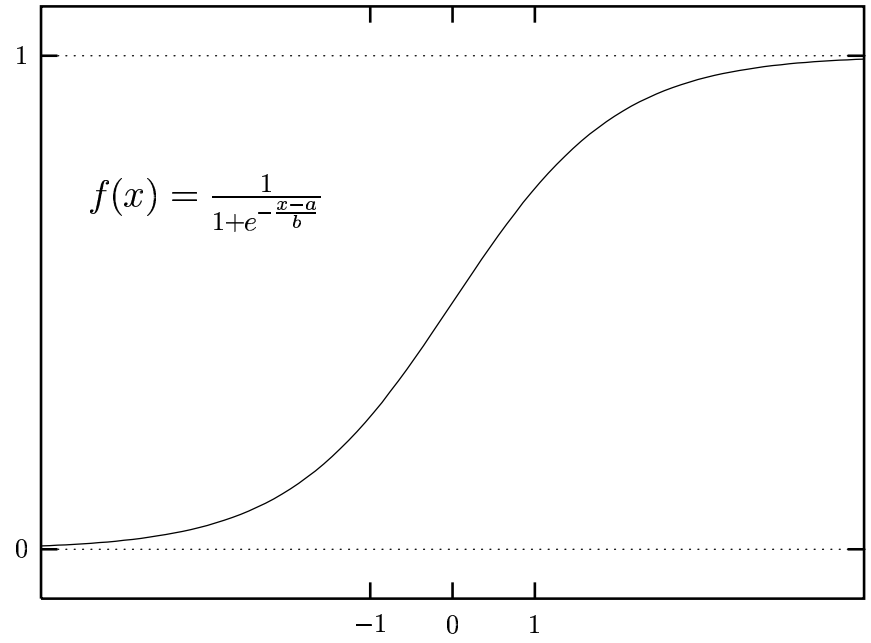
- *one of N* { `Rojo=(1,0,0), Verde=(0,1,0), Azul=(0,0,1)`}

  Aumenta en forma prohibitiva el tamaño de la red haciendo más difícil su entrenamiento.

Hay que buscar en cada caso la manera en que mejor se adapte a la información que maneje la NN.

Entrada / Salida en NNs Cuantitativas

Generalmente para el manejo de las entradas se elige una función que pueda aceptar cualquier valor y que produzca una salida en un rango limitado para evitar saturación en la NN. Una función muy utilizada es la logística. Es muy importante tener en cuenta el escalamiento.



Aprendizaje

El aprendizaje en las NNs generalmente consiste en la modificación de los *pesos* y y se pueden considerar todas la reglas de aprendizaje como variantes de la regla propuesta por Hebb *Hebbian learning rule* en 1949.

Básicamente la regla consiste en fortalecer aquellas sinapsis que unen a dos neuronas (i y j) si estas se activan simultaneamente. Por ejemplo, la versión no supervisada puede ser

$$\Delta W_{ij} = \gamma X_i X_j$$

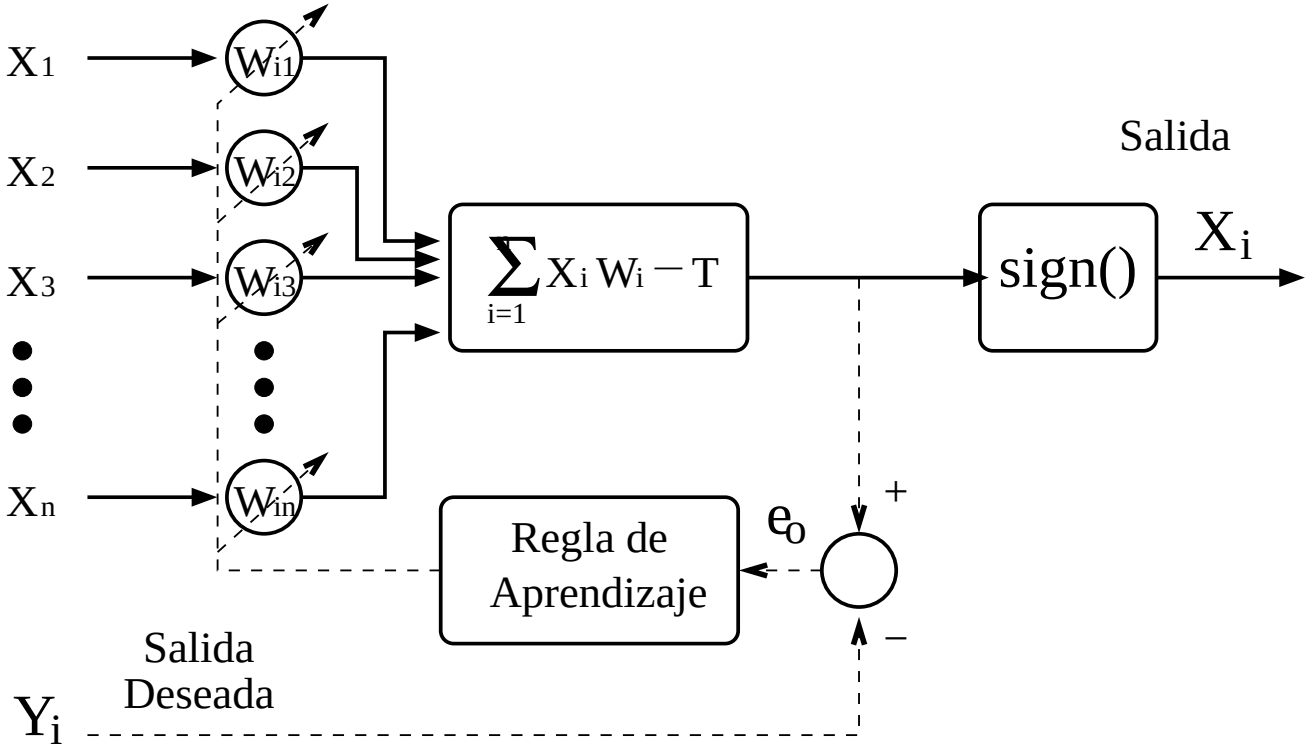
y la supervisada

$$\Delta W_{ij} = \gamma (X_i - Y_i) X_j$$

donde, $\gamma > 0$ representa la tasa de aprendizaje y Y_i la salida deseada suministrada por el supervisor.

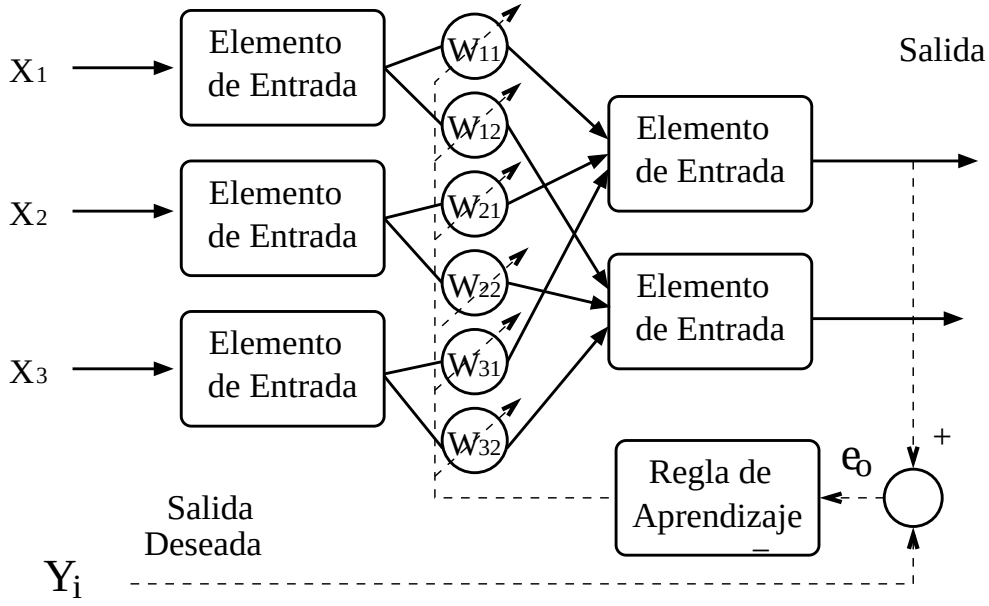
Adaline *Adaptative Linear Element*

Es el sistema más siple y consiste en un sumador lineal de las entradas $X_j \in \{-1, 1\}$ ponderadas por los *pesos* W_{ij}. A la salida del sumador hay una función signo que retorna -1 o $+1$ dependiendo del signo de las suma.



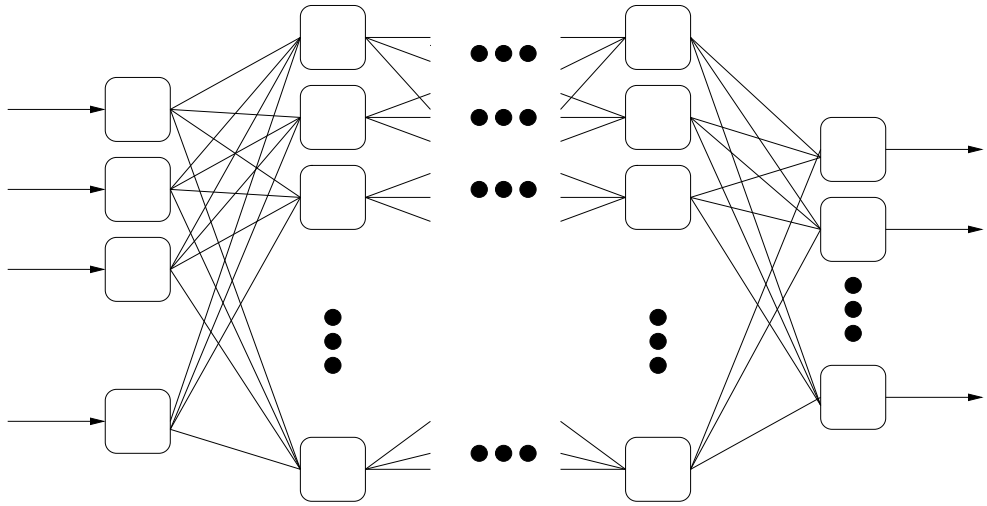
Perceptrón

El Perceptrón al igual que Adaline básicamente es un sumador lineal de entradas que cambia la entrada bipolar $X_j \in \{-1, 1\}$ por una entrada binaria $X_j \in \{0, 1\}$ ponderadas. Consiste en una capa de entrada de N elementos que se conectan con los M elementos de salida. Entre las capa de entrada y la de salida se encuentran los *pesos*



Madaline y Perceptron Multicapa (MLP)

En 1969 Minsky y Papert demostraron que con el Perceptron con una capa de entrada y otra de salida *no es posible* entrenar a la NN para comportarse como una compuerta XOR. Para resolver este tipo de problemas le propuso una topología Multicapa.



Con una capa de entrada, una o varias ocultas y una de salida.

Aprendizaje en NN multicapa

Aplicando alguna de las generalizaciones de las reglas Hebbianas, el aprendizaje comprende un proceso iterativo de dos pasos:

- Se presenta una entrada a la NN y se calcula la salida que se compara con el valor deseado, obteniendose el *error*
- El error es enviado hacia atrás *back-propagation* para calcular el cambio de cada uno de los pesos.

Hay que hacerlo de este modo ya que no se conoce cual debe ser la salida deseada de los elementos que se encuentran en las capas ocultas *hidden*

Introducción a los EAs

Los algoritmos evolutivos (EAs) son sistemas que utilizan procesos evolutivos como elementos dominantes en su diseño. Una variedad de algoritmos evolutivos se ha propuesto. Los principales EAs son:

- **Algoritmos Genéticos**
- **Estrategias Evolutivas**
- **Programación Evolutiva**
- **Sistemas Clasificadores**
- **Programación Genética**

Simulan la evolución de un grupo de individuos mediante la *Selección*, *Mutación* y *Reproducción*. Los desenvolvimiento del los EAs depende del desempeño de las estructuras individuales según lo defina el ambiente.

Que inspira el uso de EAs

– Tenemos un problema. Bueno, vamos a intentar solucionarlo. ¿Sabemos cómo hacerlo?

– No, pero ¿podemos generar varias soluciones válidas?

– Si, pero van a ser todas muy malas soluciones. Bueno, no importa. Las generamos: 40, 100, las que sean. ¿Alguna es mejor que otra? Todas son malas, pero unas son menos malas que otras. Cogemos las mejores, las otras las eliminamos. ¿Y ahora qué?

– Podemos tomar de dos en dos y mezclarlas a ver si sale algo bueno, a veces funciona. Vamos a hacerlo otra vez. Y otra.

– ¡Oye esto se estanca, ahora son todas iguales! Vamos a coger de vez en cuando alguna de las malas, no sólo de las buenas.

– ¿Y si hacemos pequeños cambios al azar a ver si mejora?

Manuel de la Herrán Gascón

Bases Biológicas de los EAs

Los EAs se basan en las ideas de:

- **Darwin**: Aquellos individuos que posean los *carácteres más ventajosos* dejarán más *descendencia*; y si tales carácteres se deben a diferencias *genéticas* tenderá a cambiar la *composición genética* de la población, aumentando el número de individuos con dichas características.
- **Malthus** Como los seres vivos se reproducen en forma geométrica y los recursos en forma aritmética, nacerán muchos más individuos de los que es posible que sobrevivan; en consecuencia si un individuo actúa de un *modo provechoso* para él, tendrá una mayor probabilidad de *sobrevivir*, y será *seleccionado* naturalmente.

Conceptos básicos de los EAs

En los EAs se establece una relación entre cada una de las posibles soluciones del problema, y un código genético, patrón o cadena de información, que de alguna forma las representa.

Partiendo de una población inicial de individuos, cada uno con un código genético propio, el EAs crea nuevos códigos genéticos.

Para ello el código genético podrá sufrir mutaciones, transmitirse a la descendencia y combinarse con otros mediante reproducción sexual. El código es seleccionado aleatoriamente decidiendo si se mantiene o no en el sistema o puede ser evaluado cuantitativamente, asignándole un peso.

Problemas para resolver con EAs

Básicamente los EAs se utilizan para buscar soluciones óptimas o cuasiótimas a problemas de alta dimensionalidad. Las posibles soluciónes del problema deben ser cualitativamente comparables. Para ello es necesario:

- Definir una estructura de datos que pueda contener patrones que representen todas las posibles solución óptima buscada.
- Definir un tipo de patrón tal que la selección no sea debido a la interacciones de los distintos segmentos del patrón, sino que existan segmentos que independientemente provocan una selección positiva.
- Definir una función de evaluación que selecciónе los mejores individuos.

Condiciones para el uso de EAs

Las condiciones que debe cumplir un problema para ser abordable son:

- El espacio de búsqueda deber ser acotado.
- Debe existir un procedimiento relativamente rápido que asigne un grado de utilidad a cada solución propuesta, de forma que corresponda directamente con la calidad de la solución o corresponda con un valor de calidad relativo al resto de la población.
- Debe existir un método de codificación de soluciones que permita: los cruzamientos para combinar las características positivas de ambos progenitores; y las mutaciones para explorar soluciones muy dispares respecto de la solución sin mutar.

Funcionamiento de los EAs

```
inicializar el contador de iteraciones
INICIALIZAR la población
PARA CADA individuo
|     EVALUAR el ajuste del individuo
|     actualizar criterio de finalización
MIENTRAS (No se cumpla el criterio de finalización)
|     incrementar el contador de iteraciones
|     SELECCIONAR a los nuevos padres
|     RECOMBINAR a los padres
|     MUTAR a la los individuos recombinados
|     PARA CADA individuo recombinado y mutado
|     |     EVALUAR el ajuste del individuo
|     |     actualizar criterio de finalización
|     SELECCIONAR a la nueva población de individuos
```

Selección de los padres

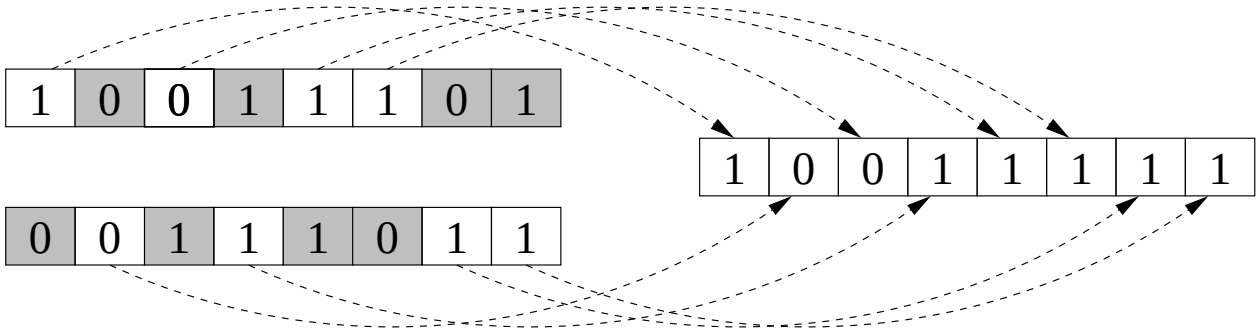
Luego de *EVALUAR* a la población hay que *SELECCIONAR* aquellos individuos que se reproducirán.

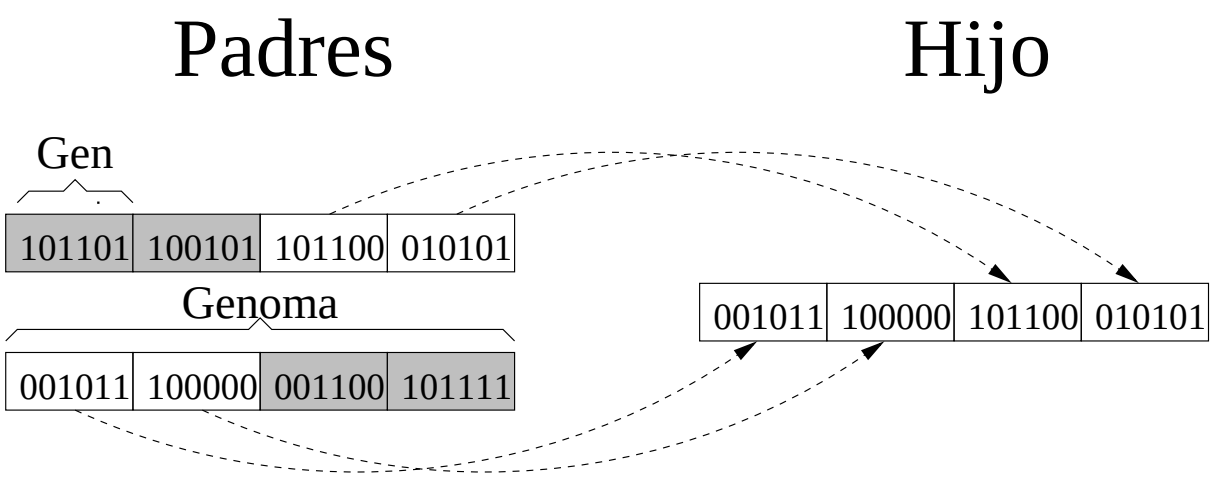
- Aleatoriamente
- Solamente los mejores (umbral)
- Con probabilidad proporcional a la evaluación
- Los m mejores y los n peores
- Todos
- Solamente el mejor

Una vez seleccionados los padres, se le aplican los diferentes operadores genéticos: Recombinación, Mutación y Translocación.

Recombinación Binaria

La recombinación se utiliza en los Algoritmos Genéticos pero no en la Programación Evolutiva ni en la Evolución de Estrategias.





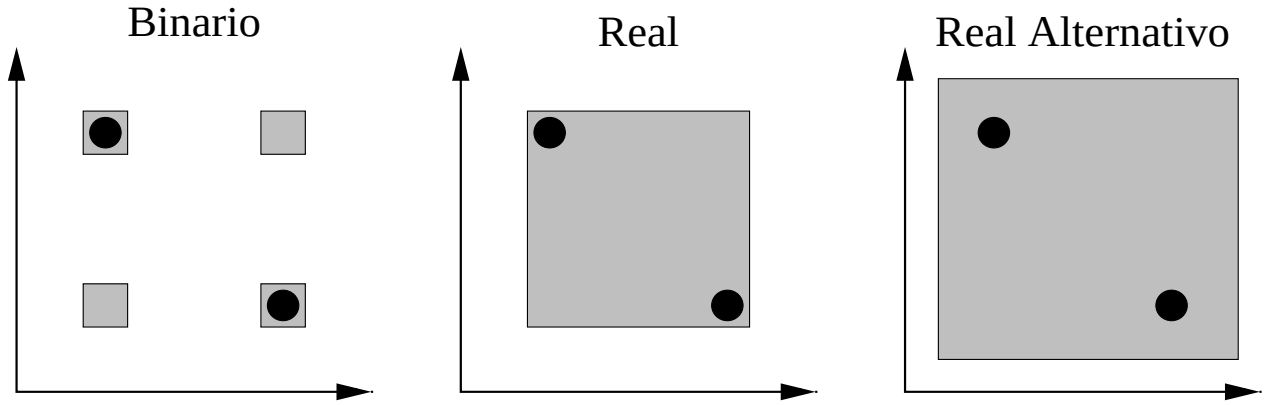
Recombinación Real

Se genera un número aleatorio $r \in [0,1]$ combinando a los padres segun:

$$\mathbf{G}_{hijo} = r\mathbf{G}_{padre1} + (1-r)\mathbf{G}_{padre2} \ ;$$

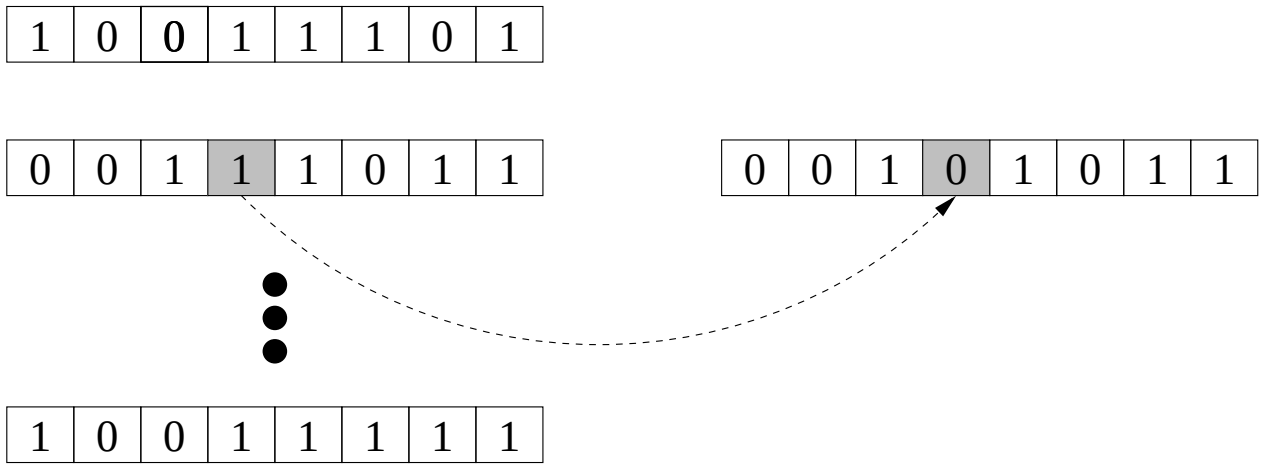
o

$$\mathbf{G}_{hijo} = \frac{|\mathbf{G}_{padre1} - \mathbf{G}_{padre2}|}{2} + \left|\frac{3}{4}(2r-1)(\mathbf{G}_{padre1} - \mathbf{G}_{padre2})\right| \ ;$$



Mutación

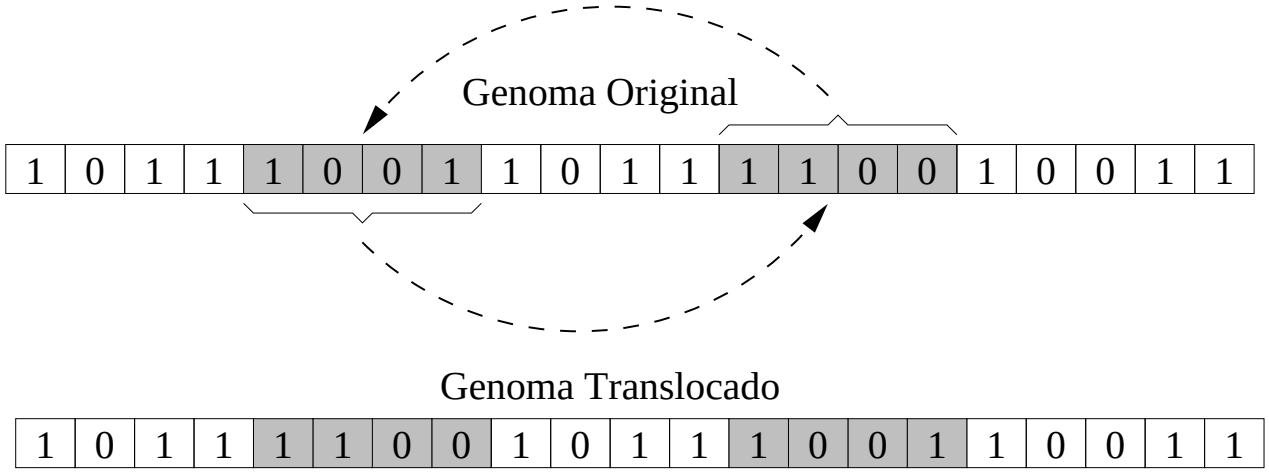
La mutación altera en forma aleatoria a algunos individuos de la nueva generación. Este operador evita que la población quede atrapada en un máximo o mínimo local.



Generalmente no se permiten mutaciones sobre los mejores individuos.

Translocación

La translocación permite que en forma aleatoria los genes de algunos individuos de la nueva generación se reordenen dentro de su genoma.



Este operador es útil cuando la información codificada dentro del genoma debe conservar ciertas propiedades que se pierden si se muta o se recombina.

Selección de la nueva población

Generalmente el tamaño de la población se mantiene constante y para esto usualmente se aplica:

- p, c: Los p padres generaron c hijos.
  De los c hijos se selecciona la nueva generación
- $p + c$: Los p padres generaron c hijos.
  De los p padres mas los c hijos se selecciona la nueva generación

Por ejemplo: $1 + 1$ significa que se selecciona a un único padre del cual por mutación o translocación se obtiene un hijo. Si el hijo es mejor que el padre el se convierte en la nueva generación.

Los criterios de Selección son similares a los de selección de los padres; pero en un **AE** no es necesario que las dos selecciones se basen en los mismos criterios.