Algoritmos genéticos

Tecnología Digital V: Diseño de Algoritmos
Universidad Torcuato Di Tella

UNIVERSIDAD
TORCUATO DI TELLA

Algoritmos genéticos

Tecnología Digital V: Diseño de Algoritmos
Universidad Torcuato Di Tella

UNIVERSIDAD
TORCUATO DI TELLA

Algoritmos exactos vs. heurísticas

- Dado un problema NP-completo, >qué alternativas tenemos?

	Algoritmos exactos	Algoritmos heurísticos
Calidad de la solución	"Buena"	"Mala"
Complejidad	"Mala"	"Buena"

Algoritmos exactos vs. heurísticas

- Dado un problema NP-completo, >qué alternativas tenemos?

	Algoritmos exactos	Algoritmos heurísticos
Calidad de la solución	"Buena"	"Mala"
Complejidad	"Mala"	"Buena"

- >Podemos tener en un mismo algoritmo una buena complejidad y una buena calidad de solución? No! A menos que P = NP.

Algoritmos exactos vs. heurísticas

- Dado un problema NP-completo, >qué alternativas tenemos?

	Algoritmos exactos	Algoritmos heurísticos
Calidad de la solución	"Buena"	"Mala"
Complejidad	"Mala"	"Buena"

- >Podemos tener en un mismo algoritmo una buena complejidad y una buena calidad de solución? No! A menos que P = NP.

1. Podemos tener algoritmos exactos con tiempos razonables,

Algoritmos exactos vs. heurísticas

- Dado un problema NP-completo, >qué alternativas tenemos?

	Algoritmos exactos	Algoritmos heurísticos
Calidad de la solución	"Buena"	"Mala"
Complejidad	"Mala"	"Buena"

- >Podemos tener en un mismo algoritmo una buena complejidad y una buena calidad de solución? No! A menos que P = NP.

1. Podemos tener algoritmos exactos con tiempos razonables,
2. o bien heurísticas con la mejor calidad de solución posible.

Algoritmos genéticos

- Los algoritmos genéticos son una técnica que se encuadra dentro del segundo grupo de algoritmos.

Algoritmos genéticos

- Los algoritmos genéticos son una técnica que se encuadra dentro del segundo grupo de algoritmos.

    1. Son algoritmos heurísticos (no garantizan la solución exacta).

Algoritmos genéticos

- Los algoritmos genéticos son una técnica que se encuadra dentro del segundo grupo de algoritmos.

  1. Son algoritmos heurísticos (no garantizan la solución exacta).
  2. Son algoritmos eficientes.

Algoritmos genéticos

- Los algoritmos genéticos son una técnica que se encuadra dentro del segundo grupo de algoritmos.

  1. Son algoritmos heurísticos (no garantizan la solución exacta).
  2. Son algoritmos eficientes.

- Están inspirados en la teoría de evolución de las especies con selección natural (la idea general se denomina bio-inspired computing).

Algoritmos genéticos

- Los algoritmos genéticos son una técnica que se encuadra dentro del segundo grupo de algoritmos.
  1. Son algoritmos heurísticos (no garantizan la solución exacta).
  2. Son algoritmos eficientes.

- Están inspirados en la teoría de evolución de las especies con selección natural (la idea general se denomina bio-inspired computing).

- Un algoritmo genético simula una población de individuos (soluciones para el problema) que va evolucionando hasta encontrar una buena solución.

Ejemplo

- Supongamos que queremos implementar un algoritmo genético para el problema de la mochila.

Ejemplo

- Supongamos que queremos implementar un algoritmo genético para el problema de la mochila.
- **Datos de entrada:**

Ejemplo

- Supongamos que queremos implementar un algoritmo genético para el problema de la mochila.
- **Datos de entrada:**
  1. Capacidad $C \in \mathbb{R}_+$ de la mochila (peso máximo).

Ejemplo

- Supongamos que queremos implementar un algoritmo genético para el problema de la mochila.
- **Datos de entrada:**
  1. Capacidad $C \in \mathbb{R}_+$ de la mochila (peso máximo).
  2. Cantidad $n \in \mathbb{N}$ de objetos.

Ejemplo

- Supongamos que queremos implementar un algoritmo genético para el problema de la mochila.
- **Datos de entrada:**

  1. Capacidad $C \in \mathbb{R}_+$ de la mochila (peso máximo).
  2. Cantidad $n \in \mathbb{N}$ de objetos.
  3. Peso $p_i \in \mathbb{R}_+$ del objeto i, para $i = 1, \ldots, n$.

Ejemplo

- Supongamos que queremos implementar un algoritmo genético para el problema de la mochila.
- **Datos de entrada:**

  1. Capacidad $C \in \mathbb{R}_+$ de la mochila (peso máximo).
  2. Cantidad $n \in \mathbb{N}$ de objetos.
  3. Peso $p_i \in \mathbb{R}_+$ del objeto i, para $i = 1, \ldots, n$.
  4. Beneficio $b_i \in \mathbb{R}_+$ del objeto i, para $i = 1, \ldots, n$.

Ejemplo

- Supongamos que queremos implementar un algoritmo genético para el problema de la mochila.
- **Datos de entrada:**
  1. Capacidad $C \in \mathbb{R}_+$ de la mochila (peso máximo).
  2. Cantidad $n \in \mathbb{N}$ de objetos.
  3. Peso $p_i \in \mathbb{R}_+$ del objeto i, para $i = 1, \ldots, n$.
  4. Beneficio $b_i \in \mathbb{R}_+$ del objeto i, para $i = 1, \ldots, n$.
- **Problema:** Determinar qué objetos debemos incluir en la mochila sin excedernos del peso máximo C, de modo tal de maximizar el beneficio total entre los objetos seleccionados.

Ejemplo

- Un candidato a solución está representado por un subconjunto de los elementos (puede cumplir el peso máximo o no).

Ejemplo

- Un candidato a solución está representado por un subconjunto de los elementos (puede cumplir el peso máximo o no).
- En un algoritmo genético, representamos una solución por medio de una secuencia de bits, denominados un cromosoma o individuo.

Ejemplo

- Un candidato a solución está representado por un subconjunto de los elementos (puede cumplir el peso máximo o no).
- En un algoritmo genético, representamos una solución por medio de una secuencia de bits, denominados un cromosoma o individuo.
- Para el problema de la mochila podemos tener n bits, de modo tal que el i-ésimo bit sea 1 si el objeto i está en el conjunto y 0 en caso contrario.

00010101101010010 10

Algoritmos genéticos

- Inicialmente tenemos una población conformada por varios individuos generados aleatoriamente.

Algoritmos genéticos

- Inicialmente tenemos una población conformada por varios individuos generados aleatoriamente.
- En cada paso de la simulación:

Algoritmos genéticos

- Inicialmente tenemos una población conformada por varios individuos generados aleatoriamente.
- En cada paso de la simulación:
  1. Algunos individuos mutan espontáneamente.

Algoritmos genéticos

- Inicialmente tenemos una población conformada por varios individuos generados aleatoriamente.
- En cada paso de la simulación:
  1. Algunos individuos mutan espontáneamente.
  2. Algunos pares de individuos se combinan, generando individuos hijos.

Algoritmos genéticos

- Inicialmente tenemos una población conformada por varios individuos generados aleatoriamente.
- En cada paso de la simulación:
  1. Algunos individuos mutan espontáneamente.
  2. Algunos pares de individuos se combinan, generando individuos hijos.
  3. Los peores individuos se eliminan y se reemplazan por nuevos.

Algoritmos genéticos

- Inicialmente tenemos una población conformada por varios individuos generados aleatoriamente.
- En cada paso de la simulación:
  1. Algunos individuos mutan espontáneamente.
  2. Algunos pares de individuos se combinan, generando individuos hijos.
  3. Los peores individuos se eliminan y se reemplazan por nuevos.
- Para evaluar cuáles son los peores individuos, se introduce una función de fitness que mide la calidad de un individuo (en cuanto a la solución del problema).

- La mutación afecta a algunos individuos aleatoriamente:

$$\begin{array}{rl} & 01101011010110101101010110101010 \\ \rightsquigarrow & 01101011010110111101010110101010 \end{array}$$

Algoritmos genéticos

- La mutación afecta a algunos individuos aleatoriamente:

$$\begin{array}{rl} & 01101011010110101101010110101010 \\ \rightsquigarrow & 01101011010110111101010110101010 \end{array}$$

- La recombinación genera dos nuevos individuos hijos a partir de dos individuos padres:

$$\begin{array}{l} 01101011010110101101010110101010 \\ 11101101111101001011101010101 \end{array}$$

Algoritmos genéticos

- La mutación afecta a algunos individuos aleatoriamente:

$$\begin{array}{rl} & 01101011010110101101010110101010 \\ \rightsquigarrow & 01101011010110111101010110101010 \end{array}$$

- La recombinación genera dos nuevos individuos hijos a partir de dos individuos padres:

$$\begin{array}{l} 01101011010110101101010110101010 \\ 11101101111101001011101010101 \end{array}$$

Algoritmos genéticos

- La mutación afecta a algunos individuos aleatoriamente:

$$\begin{array}{rl} & 01101011010110101101010110101010 \\ \rightsquigarrow & 01101011010110111101010110101010 \end{array}$$

- La recombinación genera dos nuevos individuos hijos a partir de dos individuos padres:

$$\begin{array}{rl} & 01101011010110101101010110101010 \\ & 11101101111101001011101010101 \\ \rightsquigarrow & 01101011010110101011101010101 \\ & 11101101111101001101010110101010 \end{array}$$

Algoritmos genéticos

- Con estos elementos, podemos plantear el esquema de un algoritmo genético:

```
Generar la poblacion P aleatoriamente;
while( la poblacion no es satisfactoria )
{
  Seleccionar algunos individuos de P y mutarlos;
  Seleccionar algunos pares de individuos de P y recombinarlos;
  Eliminar de P los peores individuos segÃºn la funcion de fitness;
  Reemplazar los individuos eliminados con nuevos individuos aleatorios;
}
return el mejor individuo de P;
```

Algoritmos genéticos

- Para plantear un algoritmo genético, es necesario especificar ...

Algoritmos genéticos

- Para plantear un algoritmo genético, es necesario especificar ...
  1. Cómo se codifica una solución con secuencias de bits.

Algoritmos genéticos

- Para plantear un algoritmo genético, es necesario especificar ...

  1. Cómo se codifica una solución con secuencias de bits.
  2. Una función de fitness adecuada para el problema.

Algoritmos genéticos

- Para plantear un algoritmo genético, es necesario especificar ...
  1. Cómo se codifica una solución con secuencias de bits.
  2. Una función de fitness adecuada para el problema.
- Además, un algoritmo genético tiene varios parámetros:

Algoritmos genéticos

- Para plantear un algoritmo genético, es necesario especificar ...
  1. Cómo se codifica una solución con secuencias de bits.
  2. Una función de fitness adecuada para el problema.
- Además, un algoritmo genético tiene varios parámetros:
  1. La cantidad m de individuos en la población.

Algoritmos genéticos

- Para plantear un algoritmo genético, es necesario especificar ...
  1. Cómo se codifica una solución con secuencias de bits.
  2. Una función de fitness adecuada para el problema.
- Además, un algoritmo genético tiene varios parámetros:
  1. La cantidad m de individuos en la población.
  2. La tasa de mutación t_m (probabilidad de seleccionar un individuo para mutarlo).

Algoritmos genéticos

- Para plantear un algoritmo genético, es necesario especificar ...

  1. Cómo se codifica una solución con secuencias de bits.
  2. Una función de fitness adecuada para el problema.

- Además, un algoritmo genético tiene varios parámetros:

  1. La cantidad m de individuos en la población.
  2. La tasa de mutación t_m (probabilidad de seleccionar un individuo para mutarlo).
  3. La tasa de recombinación t_r (probabilidad de seleccionar un individuo para recombinarlo con otro).

Algoritmos genéticos

- Para plantear un algoritmo genético, es necesario especificar ...
  1. Cómo se codifica una solución con secuencias de bits.
  2. Una función de fitness adecuada para el problema.
- Además, un algoritmo genético tiene varios parámetros:
  1. La cantidad m de individuos en la población.
  2. La tasa de mutación t_m (probabilidad de seleccionar un individuo para mutarlo).
  3. La tasa de recombinación t_r (probabilidad de seleccionar un individuo para recombinarlo con otro).
- Valores típicos para estos parámetros pueden ser $m = 1000$, $t_m = 0.001$ y $t_r = 0.7$, pero se deben ajustar empíricamente.

Algoritmos genéticos

- >Cuándo termina un algoritmo genético?

Algoritmos genéticos

- >Cuándo termina un algoritmo genético?

    1. Cuando se llega a un máximo de iteraciones.

Algoritmos genéticos

- >Cuándo termina un algoritmo genético?

  1. Cuando se llega a un máximo de iteraciones.
  2. Cuando la mejor solución tiene un fitness por encima de cierto umbral establecido de antemano.

Algoritmos genéticos

- >Cuándo termina un algoritmo genético?

  1. Cuando se llega a un máximo de iteraciones.
  2. Cuando la mejor solución tiene un fitness por encima de cierto umbral establecido de antemano.
  3. Cuando pasa una cierta cantidad de iteraciones sin mejora.

Algoritmos genéticos

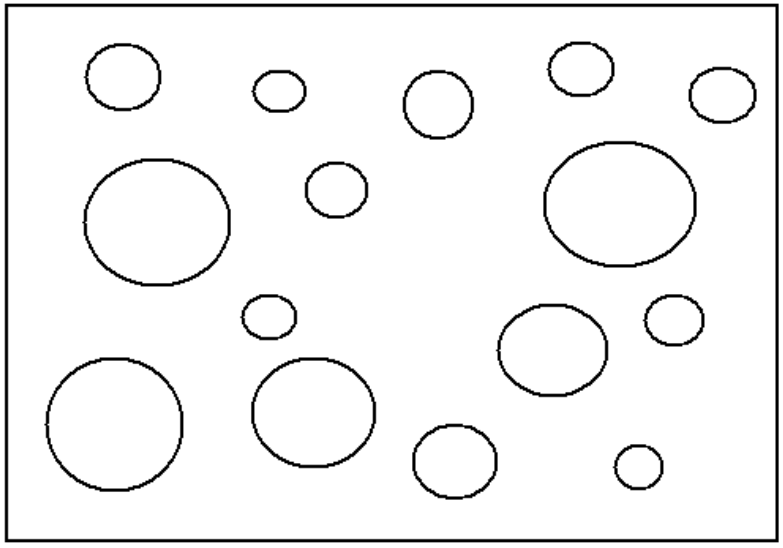
- >Cuándo termina un algoritmo genético?

  1. Cuando se llega a un máximo de iteraciones.
  2. Cuando la mejor solución tiene un fitness por encima de cierto umbral establecido de antemano.
  3. Cuando pasa una cierta cantidad de iteraciones sin mejora.

- En general, no hay un criterio definido. Un algoritmo genético es un proceso iterativo que se espera que sea convergente a la solución óptima, pero no es posible saber cuándo llega a la solución óptima.

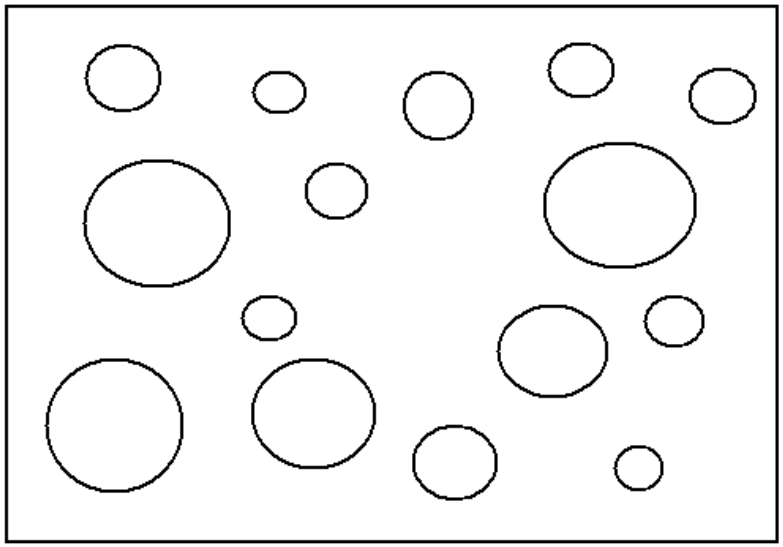
Otro ejemplo

- Dado un conjunto de círculos en el plano, encontar el círculo más grande que se puede dibujar sin tocar los círculos existentes.



Otro ejemplo

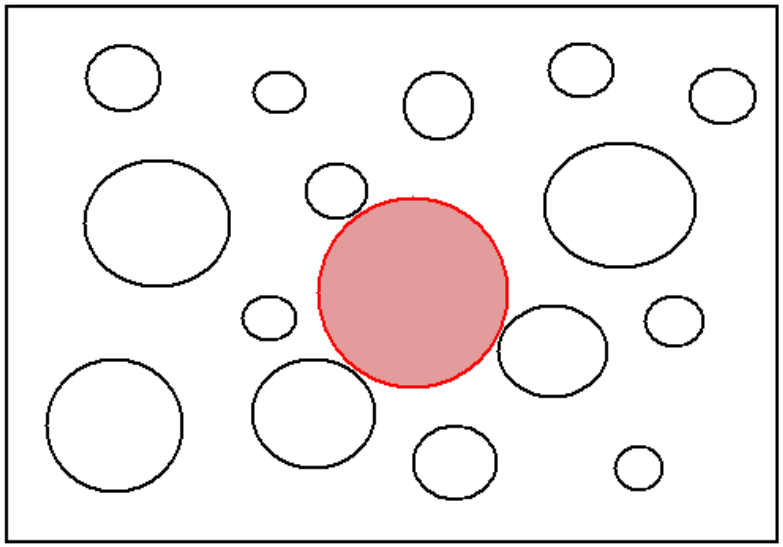
- Dado un conjunto de círculos en el plano, encontar el círculo más grande que se puede dibujar sin tocar los círculos existentes.



- Se puede resolver en forma eficiente en el plano (*largest empty circle*), pero en más dimensiones tiene complejidad exponencial.

Otro ejemplo

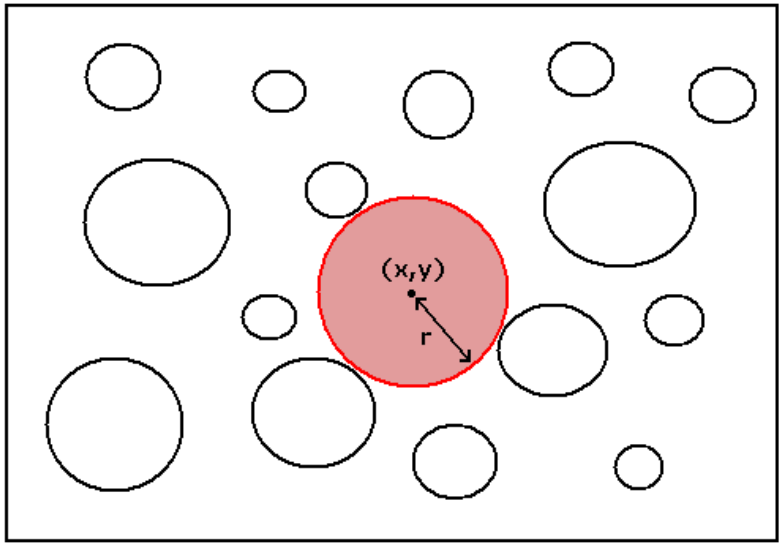
- Dado un conjunto de círculos en el plano, encontar el círculo más grande que se puede dibujar sin tocar los círculos existentes.



- Se puede resolver en forma eficiente en el plano (*largest empty circle*), pero en más dimensiones tiene complejidad exponencial.

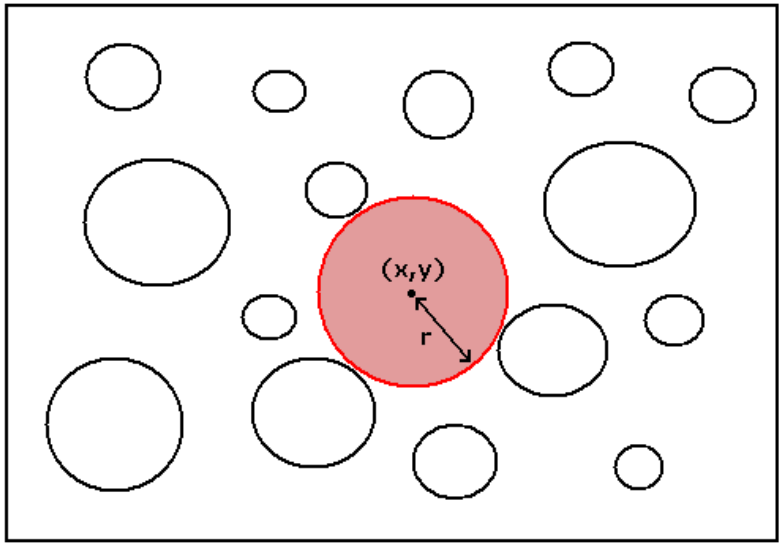
Otro ejemplo

- Una solución está representada por tres **double**s: el centro (x, y) y el radio r del círculo.



Otro ejemplo

- Una solución está representada por tres **double**s: el centro (x, y) y el radio r del círculo.



- Dados estos tres valores, podemos verificar eficientemente si corresponde a una solución válida o no.

Otro ejemplo

- Si usamos k bits para representar cada double, un cromosoma tiene $3k$ bits, y los podemos organizar de la siguiente forma:

$$\underbrace{0110101101}_{x(k \text{ bits})}\ \underbrace{0110101101}_{y(k \text{ bits})}\ \underbrace{0101101010}_{r(k \text{ bits})}$$

Otro ejemplo

- Si usamos k bits para representar cada double, un cromosoma tiene $3k$ bits, y los podemos organizar de la siguiente forma:

$$\underbrace{0110101101}_{x(k \text{ bits})}\ \underbrace{0110101101}_{y(k \text{ bits})}\ \underbrace{0101101010}_{r(k \text{ bits})}$$

- Cada una de estas tres partes se denomina un gen, y juntas conforman un cromosoma o individuo.

Otro ejemplo

- Si usamos k bits para representar cada double, un cromosoma tiene $3k$ bits, y los podemos organizar de la siguiente forma:

$$\underbrace{0110101101}_{x(k \text{ bits})}\ \underbrace{0110101101}_{y(k \text{ bits})}\ \underbrace{0101101010}_{r(k \text{ bits})}$$

- Cada una de estas tres partes se denomina un gen, y juntas conforman un cromosoma o individuo.

- La función de fitness puede ser el área del círculo menos la cantidad de círculos del input que interseca.