irenchuchu
www.wuolah.com/student/irenchuchu

TEMA 3 (II).pdf

APUNTES PARCIAL II

2º Inteligencia Artificial

Grado en Ingeniería Informática

Escuela Técnica Superior de Ingenierías Informática y de Telecomunicación
UGR - Universidad de Granada

INTELIGENCIA ARTIFICIAL: CONTINUACIÓN DEL TEMA 3

~ Búsqueda con información ~

Heurísticas

- Si se tiene conocimiento perfecto: algoritmo exacto.
- Si no se tiene conocimiento: búsqueda sin información.
- En la mayor parte de los problemas que resuelven los humanos, se está en posiciones intermedias.
- **Heurística:** conocimiento parcial sobre un problema/dominio que permite resolver problemas eficientemente en ese problema/dominio. El agente tiene conocimiento del entorno que le rodea.
- Las **heurísticas** son criterios, métodos o principios para decidir cuál de entre varias acciones promete ser la mejor para alcanzar una determinada meta.
- En IA, entendemos por **heurísticas** un método para resolver problemas que en general no garantiza la solución óptima, pero que en media produce resultados satisfactorios en la resolución de un problema.
- Una heurística encapsula el conocimiento específico/experto que se tiene sobre un problema, y sirve de guía para que un algoritmo de búsqueda pueda encontrar una solución válida aceptable.
- Eventualmente, una heurística puede devolver siempre soluciones óptimas bajo ciertas condiciones (requiere demonstración).

8-puzzle:

- En IA, implementaremos heurísticas como funciones que devuelven un valor numérico, cuya maximización o minimización guiará al proceso de búsqueda a la solución.
- Ejemplo de heurística en el problema de 8-puzzle:
  - **f(n)** = nº de fichas descolocadas en comparación con la posición objetivo a alcanzar.
  - **Objetivo:** Minimizar **f**.
  - **n** es un estado (posición de las piezas) del problema.
  - **f(n)** es la *función heurística*.

f((2, 8, 3, 1, 6, 4, 7, 0, 5))= 9

Métodos de escalada.

Si dibujamos las soluciones como puntos en el espacio, una búsqueda local consiste en seleccionar la solución mejor en vecindario de una solución inicial, e ir viajando por las soluciones del espacio hasta encontrar un óptimo (local o global).

E: Estado activo.

Algoritmo de escalada simple:

```
while (E no sea el objetivo
        y queden nodos por explorar a partir de E) {
        Seleccionar operador A para aplicarlo a E
        Evaluar f(A(E))
        if (f(A(E))>f(E)) {
                E = R(E)
        }
}
```

Algoritmo de escalada por la máxima pendiente:

```
while (queden nodos por explorar a partir de E) {
        Para todos los operadores A_i, obtener E_i=A_i (E)
        Evaluar f(E_i) para todos los estados E_i=A_i (E)
        Seleccionar Emax tal que f(E_max) = max{f(E_i)}
        if (f(E_max)>f(E)) {
            E = E_max
        } else return E
}
```

Características:

- **Completitud:** no tiene por qué encontrar la solución.
- **Admisibilidad:** no siendo completo, aun menos será admisible.
- **Eficiencia:** rápido y útil si la función es monótona (de)creciente.

Algunas variaciones estocásticas.

- Algoritmo de escalada estocástico.
- Algoritmo de escalada de primera opción.
- Algoritmo de escalada de reinicio aleatorio.
- Enfriamiento simulado.
    - Es un método de búsqueda local.
    - Se basa en principios de termodinámica.
    - Al contrario que otros métodos de ascensión de colinas, permite visitar soluciones peores que la actual para evitar óptimos locales.
    - Un poco de historia:
        - En el campo de la termodinámica, en los años 50 se simuló el proceso de enfriamiento en sistemas de partículas hasta que se llegaba a un estado estable.
        - El proceso simulaba la diferencia de energía del sistema, GAMAE, y se quería GAMAE seguía la siguiente fórmula (t es la temperatura actual del sistema; k es una constante física):

$$P[\delta E] = e^{-\frac{\delta E}{k \cdot T}}$$

    - Analogía entre el proceso de enfriamiento y el algoritmo de enfriamiento simulado:
        - Los estados por los que pasa el sistema físico de partículas equivalen a las soluciones factibles del algoritmo.
        - La energía E del estado actual del sistema es el valor de la función objetivo de la solución actual. Ambos tienen que minimizarse.
        - Un cambio de estado en el sistema equivale a explorar el entorno de una solución y viajar a una solución vecina.
        - El estado final estable (congelado) es la solución final del algoritmo.
    - La solución inicial se puede generar de forma aleatoria, por conocimiento experto, o por medio de otras técnicas algorítmicas como greedy.
    - La actualización de temperatura también es heurística, y hay varios métodos:
        - T <- INIFITO * T, con INFINITO en (0,1)
        - T <- 1/(1+k), con k = número de iteraciones del algoritmo hasta el momento, etc.
    - Número de vecinos a generar: Fijo N(T)=cte, dependiente de la temperatura N(T)=f(T), etc.

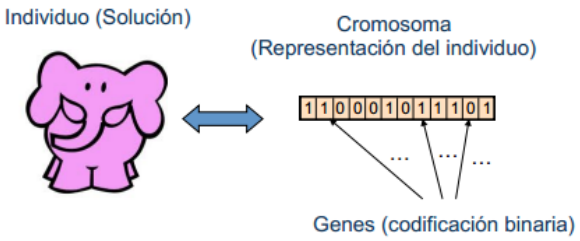
- Tanto la temperatura inicial como la temperatura final Ti y Tf son parámetros de entrada al algoritmo.
- Es difícil asignar un valor concreto a Tf, por lo que la condición de parada se suele sustituir por un número específico de iteraciones a realizar.
- Ventajas:
  - Al ser un método probabilístico, tiene capacidad para salir de óptimos locales.
  - Es eficiente.
  - Es fácil de implementar.
- Inconvenientes:
  - Encontrar la temperatura inicial Ti, el método de actualización de temperatura INFINITO, el número de vecinos a generar en cada estado y el número de iteraciones óptimo es una tarea que requiere de muchas pruebas de ensayo y error hasta que ajustamos los parámetros óptimos.
- Pese a todo, el algoritmo puede proporcionar soluciones mucho mejores que utilizando algoritmos no probabilísticos.

Algoritmos genéticos.

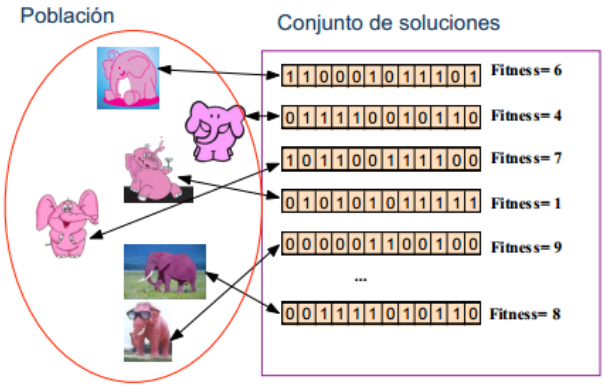
- La simulación de procesos naturales es un campo de investigación muy amplio en Inteligencia Artificial.
- Ejemplos son la computación evolutiva, biocomputación, algoritmos bioinspirados, etc.
- Si ha funcionado bien en la naturaleza, ¿porqué una simulación de estos procesos no iba a proporcionar buenos resultados en un computador?
- Ejemplos:
  - Algoritmos genéticos.
  - Algoritmos basados en Colonias de Hormigas.
  - Algoritmos basados en inteligencia de enjambres.
- Son algoritmos de optimización basados en proceso de la evolución natural de Darwin.
- En un proceso de evolución, existe una población de individuos. Los más adecuados a su entorno se reproducen y tienen descendencia (a veces con mutaciones que mejoran su idoneidad al entorno). Los más adecuados sobreviven para la siguiente generación.
- No necesitan partir de un nodo/estado inicial ya que hay toda una población.
- Su objetivo es encontrar una solución cuyo valor de función objetivo sea óptimo.

- **Cromosoma** ←→ Vector representación de una solución al problema.
- **Gen** ←→ Característica/Variable/Atributo concreto del vector de representación de una solución
- **Población** ←→ Conjunto de soluciones al problema.
- **Adecuación al entorno** ←→ Valor de función objetivo (fitness).
- **Selección natural** ←→ Operador de selección.
- **Reproducción sexual** ←→ Operador de cruce.
- **Mutación** ←→ Operador de mutación.
- **Cambio generacional** ←→ Operador de reemplazamiento.

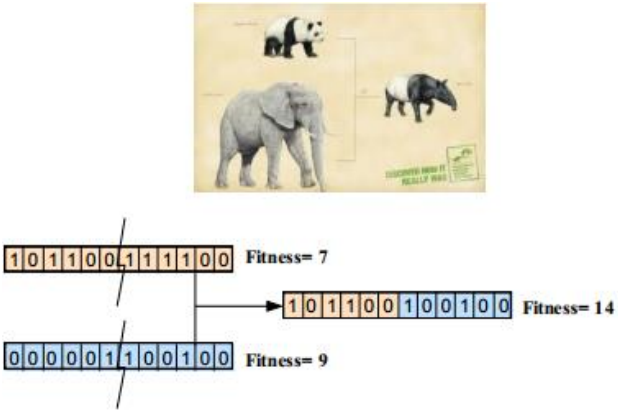
- Ejemplo:
  - **Cromosoma que codifica una solución a un problema.** Cada característica del problema es un valor 0/1.



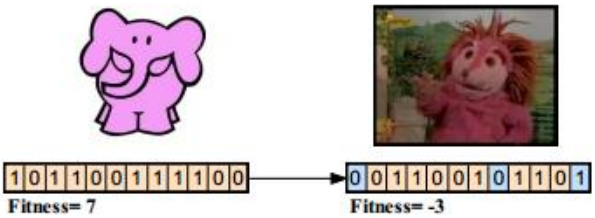
  - **Población:** Conjunto de individuos (cada uno con su fitness)



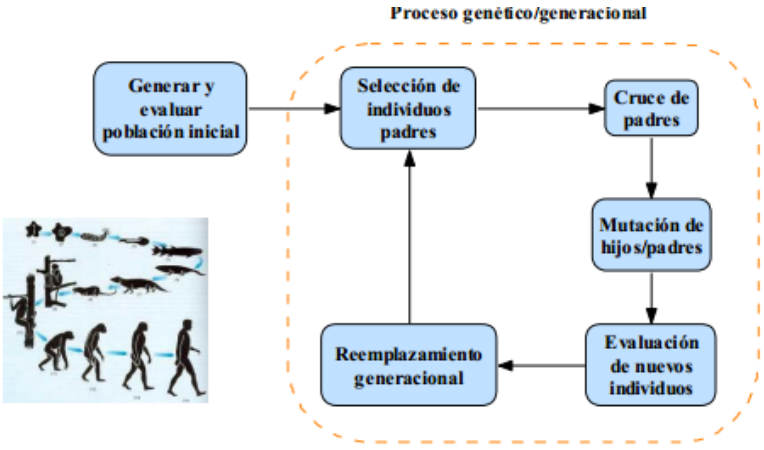
  - **Cruce:** Combinación de soluciones de la población para generar descendientes.



  - **Mutación:** Uno o más genes de un individuo pueden mutar para generar una nueva solución. En la población, hay una probabilidad dada a priori de que un individuo pueda mutar. A su vez, cuando un individuo muta, existe otra probabilidad de que cada gen mute o no.



- Proceso de algoritmo genético:



Búsqueda primero el mejor.

Se selecciona un nodo para su expansión basada en una función de evaluación. Esta devuelve un número que sirve para representar lo deseable que sería la expresión de un nodo. Se escoge el que parece ser mejor pero puede no serlo.

Algoritmo A*.

- ABIERTOS contiene el nodo inicial, CERRADOS está vacío.
- Comienza un ciclo que se repite hasta que se encuentra solución o hasta que ABIERTOS queda vacío.
    - Seleccionar el mejor nodo de ABIERTOS.
    - Si es un nodo objetivo terminar.
    - En otro caso se expande dicho nodo.
    - Para cada uno de los nodos sucesores.
        - Si está en ABIERTOS insertarlo manteniendo la información del mejor padre.
        - Si está en CERRADOS insertarlo manteniendo la información del mejor padre y actualizar la información de los descendientes.
        - En otro caso, insertarlo como un nodo nuevo.

- Características:
    - **Completitud:** si existe solución, la encuentra.
    - **Admisibilidad:** si hay una solución óptima, la encuentra si:
        - El número de sucesores es finito para cada nodo,
        - C(ni, nj) > GAMMA > 0 en cada arco, y
        - La función h(n) es admisible: h(n) <= h*(n)

- Dificultades del proceso:
    - Los procesos de percepción no siempre pueden obtener la información necesaria acerca del estado del entorno.
    - Las acciones pueden no disponer siempre de modelos de sus efectos.
    - Puede haber otros procesos físicos, u otros agentes, en el mundo.
    - En el tiempo que transcurre desde la construcción de un plan, el mundo puede cambiar de tal manera que el plan ya no sea adecuado.
    - Podría suceder que se le requiriese al agente actual antes de que pudiese completar una búsqueda de un estado objetivo.
    - Aunque el agente dispusiera de tiempo suficiente, sus recursos de memoria podrían no permitirle realizar la búsqueda de un estado objetivo.

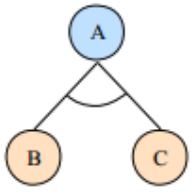
Heurísticas sobre el proceso de búsqueda:

- Búsqueda orientada a subobjetivos.
- Búsqueda con horizonte.
- Búsqueda jerárquica.

Grafo Y/O

Composición de grafos destinados a la resolución de problemas descomponibles. Se descomponen de grafos "Y" y de grafos "O", que indican el orden de tareas a realizar.

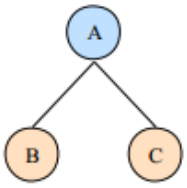
- Descomposición de problemas: arcos Y.
- Resolución de problemas: arcos O.
- Concepto de solución: subgrafo solución.
- **Grafo Y:** Para completar el objetivo/tarea A, es necesario terminar antes los objetivos/tareas B y C.



    - En el cálculo proposicional, la expresión del grafo Y anterior correspondiente sería de la siguiente forma:

$$B \cdot C \rightarrow A$$

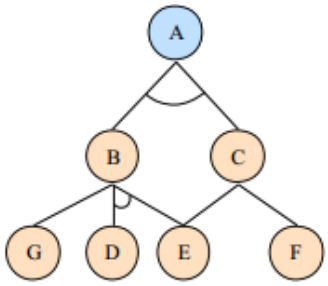
- **Grafo O:** Para completar el objetivo/tarea A, es necesario terminar antes o bien el objetivo/tarea B, o bien el objetivo/tarea C, es decir, para completar un nodo es necesario completar antes uno de sus hijos.



- En el cálculo proposicional, la expresión del grafo O anterior correspondiente sería de la siguiente forma:

$$B+C \rightarrow A$$

- **Grafo Y/O:** combinación de grafos Y, y grafos O que indican el orden de consecución de tareas a realizar para alcanzar el objetivo.



- En el cálculo proposicional, la expresión del grafo Y/O anterior correspondiente sería de la siguiente forma:

$$B \cdot C \rightarrow A;\ G+D \cdot E \rightarrow B;\ E+F \rightarrow C$$

- Un nodo que no se descompone o simplifica se llama nodo terminal.
- Para resolver un grafo Y/O, cada nodo se resuelve de la siguiente manera:
  - **Si es un nodo Y:** Resolver todos sus hijos. Combinar la solución y solucionar el nodo. Devolver su solución.
  - **Si es un nodo O:** Resolver un hijo y ver si devuelve solución. En caso contrario, resolver el siguiente hijo, etc. Cuando ya esté resuelto algún hijo, combinar la solución en el nodo y devolverla.
  - **Si es un nodo terminal:** Resolver subproblema asociado y devolverla.
- Mejora: Para seleccionar el orden de resolución de nodos hijos, se puede utilizar alguna medida de estimación del coste de resolución.

Búsqueda dirigida:

- Búsqueda de tipo A*, en la que se especifica un factor de ramificación que limita el número de nodos a expandir.

Búsqueda jerárquica:

- El agente sigue el plan generado por el algoritmo durante un corto periodo de tiempo. Cuando alcanza el final del nivel más bajo del plan, necesitará planear la siguiente acción con mayor detalle.
- Cuando el plan acaba, el algoritmo es llamado de nuevo.

Función heurística:

- Su objetivo es utilizar la información disponible para minimizar la búsqueda hasta su nodo objetivo.
- Una heurística es admisible si nunca sobreestima el costo de alcanzar el nodo objetivo.

Algoritmo de enfriamiento simulado:

- Permite visitar soluciones peores a la actual, evitando el estancamiento en óptimos locales.
- DIFICULTAD: es que requiere de muchas pruebas de ensayo y error hasta ajustar los parámetros óptimos.
- Se basa en principios de termodinámica y es un método de búsqueda local.