Volumen

2

EL PROBLEMA DEL VIAJANTE

Heurísticas

Taller de Astronomía

Autora: Profa. Ana Inés Gómez de Castro

INTRODUCCIÓN A LA ASTRONOMÍA

Taller de Astronomía

© Ana Inés Gómez de Castro
Facultad de Ciencias Matemáticas
Universidad Complutense de Madrid
email:aig@mat.ucm.es

Tabla de contenido

Capítulo

1

Combinatoria y Heurísticas

El problema del viajante

La tercera parte del ejercicio "el viajero estelar eficiente" tiene por objeto introducir a los alumnos en un problema básico de *Inteligencia Artificial* (IA) partiendo de ejercicios de combinatoria sencillos. El problema del viajante se introduce a través de las visitas a otros Sistemas Planetarios.

En un primer paso, sólo se permite viajar a 4 sistemas por lo cual es sencillo enumerar las posibles trayectorias:

$$\mathcal{P}_4 = 4! = 4*3*2*1 = 48$$

El problema se complica al permitirles viajar entre 10 de los 155 sistemas conocidos. En este caso, la simple enumeración, de las posibles trayectorias le llevaría a:

$$\mathcal{P}_{10} = 10! = 10*9*8*7*6*5*4*3*2*1 = 3.628.800$$

más de tres millones de combinaciones posibles!. Si tardaran sólo 3 segundos en representarlas con nuestra interfaz gráfica deberían, o bien emplear 12.6 días para probarlas todas, o bien tener MUCHA suerte. Este problema es una variante astronómica del "*problema del viajante*", un clásico en Investigación Operativa (IO). El problema del viajante, en sí mismo, plantea un "cierto" grado de complicación computacional puesto que el número de soluciones posibles crece exponencialmente con el número de "planetas a visitar" y no hay ningún algoritmo capaz de evitar la enumeración, explícita o implícita, de todas las soluciones[1]. El ejercicio del "*viajero estelar eficiente*" se plantea como introducción al concepto de "algoritmos heurísticos" o algoritmos capaces de ofrecer una solución

[1] Este tipo de problemas se denominan "***NP-hard***" (o problemas con grado de dificultad *NP*). La complejidad computacional es una disciplina matemática rigurosa que muestra cómo la mayor parte de los problemas de optimización pueden agruparse en clases, de tal manera, que todos los problemas de la misma clase tienen una complejidad similar. La clase de problemas "***NP-hard***" es la más importante.

suficientemente buena al problema sin tener que enumerar todas las soluciones posibles.

Se han desarrollado un gran número de heurísticas para resolver este tipo de problemas, desde algoritmos muy sencillos de re-organización de la secuencia en la que se realiza el viaje a algoritmos mucho más complejos como las redes neuronales o los algoritmos genéticos. El ejercicio que se presenta dentro de esta aplicación educativa incluye una interfaz para codificar gráficamente un algoritmo sencillo (de re-ordenación de la secuencia de viaje) que se aplica posteriormente de manera sistemática hasta encontrar la solución óptima. El ejercicio está ideado para que los alumnos puedan proponer sus propios algoritmos (si alguno es especialmente eficiente hay una utilidad para enviarlo por correo electrónico a HOU-España). En el resto de este capítulo se incluyen algunos algoritmos sencillos que pueden ser utilizados por el profesor para iniciar a los alumnos en el uso de la herramienta.

Algoritmos heurísticos para la resolución del problema del viajante

Existen diferentes tipos de heurísticas, según el modo en que buscan y construyen sus soluciones. Una posible clasificación las divide en heurísticas constructivas, y de mejoramiento iterativo.

a) ***Métodos constructivos o puros***: Consisten en ir construyendo poco a poco el camino del viajante, añadiendo vértices siguiendo un criterio concreto. Una vez conseguido el camino, ó solución factible, no lo intenta mejorar. El más popular de estos métodos lo constituyen los algoritmos "codiciosos" (greedy), que construyen paso a paso la solución buscando el máximo beneficio en cada paso.

b) ***Métodos de mejoramiento iterativo o mejora local***: Estos métodos no tratan de llegar a una solución factible, sino que parten de una de ellas (obtenida quizás mediante un método constructivo), y, mediante alteraciones de esa solución, van pasando de forma iterativa a otra mejor hasta que se cumpla un determinado criterio de fin.

Las heurísticas pueden ser simples o complejas. Los algoritmos simples tienden a tener reglas de terminación bien definidas, y se detienen en un óptimo local. Los algoritmos más complejos pueden no tener reglas de terminación estándar, y típicamente buscan soluciones mejores hasta alcanzar un punto de parada arbitrario.

Algoritmos constructivos o puros

El ***algoritmo del vecino más cercano***, utiliza una idea muy básica (quizás la primera que se nos ocurriría a cualquiera) para construir un camino factible. Construye iterativamente un camino tomando en cada paso la estrella más próxima

de las que ya están incluidas en la ruta provisional. En cada paso, el método necesita un vértice n_0 que es el de inicio, otro *t* que es en el que acaba el recorrido construido hasta el momento y un conjunto de vértices W todavía sin visitar. Inicialmente, $t = n_0$ y $W = N \setminus \{n_0\}$.

ALGORITMO 1 (*Vecino más cercano*)

Paso 0: Seleccionar un vértice arbitrario n_0. Poner $t = n_0$ y $W = N \setminus \{n_0\}$

Paso 1: Mientras $W \neq \varnothing$ hacer:

escoger $j \in W$ tal que $c_{tj} = \min\{c_{ti} \mid i \in W\}$

conectar *t* a *j* y poner $W = W \setminus \{j\}$ y $t = j$.

Paso 2: Conectar *t* al vértice inicial n_0 para formar un ciclo hamiltoniano.

Una pequeña variante de este algoritmo consiste en permitir que el camino se vaya ampliando por cualquiera de sus dos extremos, dando lugar al algoritmo del vecino bilateral más próximo.

Heurísticas de mejoramiento iterativo

La idea es ir mejorando una solución factible seleccionada al azar (por ejemplo, obtenida por el método del vecino mas próximo) intercambiando iterativamente el orden de las ciudades a visitar. Podríamos generalizarlo en los siguientes pasos:

1. Generar una solución factible inicial T
2. Intentar hallar una solución factible mejorada T', por alguna transformación de T.
3. Si se encuentra una solución mejorada (es decir, que produzca un menor valor de la función objetivo), reemplazar T por T' y repetir desde el paso 2.
4. Si no puede encontrarse una solución mejorada, T es una solución localmente óptima.

Lo que diferencia y caracteriza cada procedimiento iterativo es el paso 2; según cómo hagamos la transformación que trata de mejorar una solución dada tendremos uno u otro método.

Básicamente, la solución a la que llegan este tipo de métodos es un óptimo local, así, la existencia de dichos óptimos locales es uno de los mayores inconvenientes que presentan. Si a lo largo de la búsqueda se cae en un óptimo local, en principio la heurística no sabría continuar, pues quedaría detenida en ese punto.

Otro de los problemas a los que se enfrentan estas heurísticas es la dependencia de la solución inicial de la que se parta. Obviamente, el punto inicial tiene una gran influencia sobre la posibilidad de caer o no en un óptimo local.

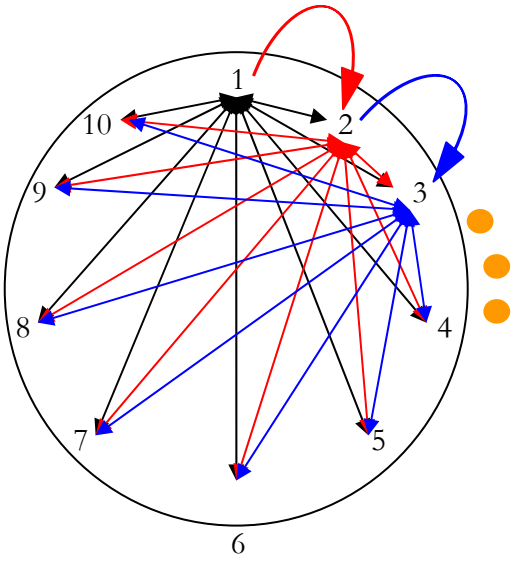
Alguna heurísticas sencillas de mejoramiento iterativo

Estos métodos parten de una solución factible y, mediante alteraciones de esa solución, van pasando de forma iterativa a otra mejor hasta que se cumpla un determinado criterio de fin. Las heurísticas pueden ser simples o complejas. Los algoritmos simples tienden a tener reglas de terminación bien definidas, y se detienen en un óptimo local. Los algoritmos más complejos pueden no tener reglas de terminación estándar, y típicamente buscan soluciones mejores hasta alcanzar un punto de parada arbitrario (p.e. los algoritmos genéticos). Lo que diferencia y caracteriza cada procedimiento iterativo es la transformación del punto 2 del esquema anterior.

A continuación se proponen tres transformaciones sencillas de intercambio: *algoritmos de Mejora Simple, de Intercambio de Mejora y de Lin-Kernighan,*. El algoritmo más sencillo es el *de Intercambio de Mejora.* En este algoritmo se reorganiza el plan de viaje intercambiando el orden en que se visitan dos estrellas. Se selecciona una primera estrella y se re-organiza el viaje intercambiándola con todas las estrellas posibles. Se evalúa la distancia en todas estas permutaciones y se selecciona la mejor. Después se selecciona otra estrella y se repite el procedimiento. De esta manera, en vez de calcular las N! permutaciones se calculan:

$$N\sum_{i=1}^{i=N-1} i$$

combinaciones posibles. Por ejemplo, en el caso de visitar 10 estrellas pasaríamos de tener que calcular las 10! permutaciones posibles a calcular sólo 10(9+8+7+6+5+4+3+2+1) =450, tal y como se indica en la figura,



Otros algoritmos se basan en realizar cambios más complejos tal y como se ilustra en la figura.